

Package ‘sybillion’

July 17, 2026

Title Official R Client for the 'Sybillion' API

Version 0.1.0

Description Wrapper and generated client for the 'Sybillion' Developers Portal API (forecasts, drivers, catalog, usage). Authenticate with an API key using 'Authorization: Bearer'. See <<https://sybillion.dev/docs/>>.

URL <https://sybillion.dev/docs/>

BugReports <https://sybillion.dev/docs/>

Depends R (>= 4.1.0)

Encoding UTF-8

License Apache License (>= 2)

Suggests testthat, roxygen2

Imports jsonlite, http2, R6, base64enc, stringr

Config/roxygen2/version 8.0.0

NeedsCompilation no

Author Sybillion [aut, cre, cph]

Maintainer Sybillion <support@sybillion.com>

Repository CRAN

Date/Publication 2026-07-17 12:50:02 UTC

Contents

AlertItemV1	2
AlertsRequestV1	6
AnyType	9
ApiClient	10
ApiResponse	15
ApiV1AlertsPost200Response	16
ApiV1DriversPost200Response	19
ApiV1DriversPost200ResponseData	22
ApiV1ForecastsIdGet200Response	25

ApiV1ForecastsPost202Response	29
ApiV1JobsGet200Response	32
ApiV1UsageGet200Response	35
AutoRechargeState	38
CategoryItemV1	42
CategoryListResponse	45
Client	48
DefaultApi	52
DEFAULT_PUBLIC_API_BASE_URL	60
DriverItemV1	60
ErrorMessage	63
EuroTranche	66
Filters	69
ForecastArtifactMeta	73
ForecastRequestV1	76
HealthGet200Response	80
HealthGet200ResponseComponentsValue	83
HealthGet503Response	86
HealthGet503ResponseComponentsValue	89
JobsPagination	92
JobSummary	96
MeResponse	99
MeResponseSignupTrial	103
NewsItemV1	106
Pagination	110
RecommendRequestV1	113
RegionItemV1	117
RegionListResponse	120
resolve_api_url	123
SYBILION_API_BASE_URL_ENV	123
sybillion_client	124
TimeseriesMetadata	124
UsageEvent	127
ValidationErrorResponse	131
ValidationErrorResponseDetailsInner	134

Index	138
--------------	------------

AlertItemV1	<i>AlertItemV1</i>
-------------	--------------------

Description

A single alert returned by the upstream Recommend service.

AlertItemV1 Class

Format

An R6Class generator object

Details

Create a new AlertItemV1

Public fields

name Human-readable name of the dataset or index that triggered the alert. character [optional]
news Related news articles driving this alert. list([NewsItemV1](#)) [optional]
pct_change Percentage change that triggered the alert (negative = decline, positive = surge). numeric [optional]
trending Whether this alert is currently trending across the platform. character [optional]

Methods**Public methods:**

- [AlertItemV1\\$new\(\)](#)
- [AlertItemV1\\$list\(\)](#)
- [AlertItemV1\\$toSimpleType\(\)](#)
- [AlertItemV1\\$fromJSON\(\)](#)
- [AlertItemV1\\$toJSONString\(\)](#)
- [AlertItemV1\\$fromJSONString\(\)](#)
- [AlertItemV1\\$validateJSON\(\)](#)
- [AlertItemV1\\$toString\(\)](#)
- [AlertItemV1\\$isValid\(\)](#)
- [AlertItemV1\\$getInvalidFields\(\)](#)
- [AlertItemV1\\$print\(\)](#)
- [AlertItemV1\\$clone\(\)](#)

[AlertItemV1\\$new\(\)](#): Initialize a new AlertItemV1 class.

Usage:

```
AlertItemV1$new(
  name = NULL,
  news = NULL,
  pct_change = NULL,
  trending = NULL,
  ...
)
```

Arguments:

name Human-readable name of the dataset or index that triggered the alert.
news Related news articles driving this alert.
pct_change Percentage change that triggered the alert (negative = decline, positive = surge).

trending Whether this alert is currently trending across the platform.
 ... Other optional arguments.

AlertItemV1\$toList(): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

`AlertItemV1$toList()`

Returns: AlertItemV1 as a base R list.

Examples:

```
# convert array of AlertItemV1 (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

AlertItemV1\$toSimpleType(): Convert AlertItemV1 to a base R type

Usage:

`AlertItemV1$toSimpleType()`

Returns: A base R type, e.g. a list or numeric/character array.

AlertItemV1\$fromJSON(): Deserialize JSON string into an instance of AlertItemV1

Usage:

`AlertItemV1$fromJSON(input_json)`

Arguments:

`input_json` the JSON input

Returns: the instance of AlertItemV1

AlertItemV1\$toJSONString(): To JSON String

Usage:

`AlertItemV1$toJSONString(...)`

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: AlertItemV1 in JSON format

AlertItemV1\$fromJSONString(): Deserialize JSON string into an instance of AlertItemV1

Usage:

`AlertItemV1$fromJSONString(input_json)`

Arguments:

`input_json` the JSON input

Returns: the instance of AlertItemV1

AlertItemV1\$validateJSON(): Validate JSON input with respect to AlertItemV1 and throw an exception if invalid

Usage:

```
AlertItemV1$validateJSON(input)
```

Arguments:

input the JSON input

AlertItemV1\$string(): To string (JSON format)

Usage:

```
AlertItemV1$string()
```

Returns: String representation of AlertItemV1

AlertItemV1\$isValid(): Return true if the values in all fields are valid.

Usage:

```
AlertItemV1$isValid()
```

Returns: true if the values in all fields are valid.

AlertItemV1\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

```
AlertItemV1$getInvalidFields()
```

Returns: A list of invalid fields (if any).

AlertItemV1\$print(): Print the object

Usage:

```
AlertItemV1$print()
```

AlertItemV1\$clone(): The objects of this class are cloneable with this method.

Usage:

```
AlertItemV1$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `AlertItemV1$toList()`
## -----

# convert array of AlertItemV1 (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

AlertsRequestV1

*AlertsRequestV1***Description**

Body of 'POST /api/v1/alerts'. 'filters.limit' controls how many alerts are returned (default ****100****, max ****1000****). 'date_from' / 'date_to' are optional date bounds (YYYY-MM-DD).

AlertsRequestV1 Class

Format

An R6Class generator object

Details

Create a new AlertsRequestV1

Public fields

context_enriched When true, treat the supplied metadata as already context-enriched. character
 date_from Optional start date bound for alert detection (YYYY-MM-DD). character [optional]
 date_to Optional end date bound for alert detection (YYYY-MM-DD). character [optional]
 filters Optional. 'limit' controls the number of alerts returned (****0–1000****, default ****100****).
 'categories[]' and 'regions[]' narrow the alert universe; each must be an integer ****1–9999****.
 Values are not verified against catalog APIs. [Filters](#) [optional]
 metadata [TimeseriesMetadata](#)

Methods**Public methods:**

- [AlertsRequestV1\\$new\(\)](#)
- [AlertsRequestV1\\$toList\(\)](#)
- [AlertsRequestV1\\$toSimpleType\(\)](#)
- [AlertsRequestV1\\$fromJSON\(\)](#)
- [AlertsRequestV1\\$toJSONString\(\)](#)
- [AlertsRequestV1\\$fromJSONString\(\)](#)
- [AlertsRequestV1\\$validateJSON\(\)](#)
- [AlertsRequestV1\\$toString\(\)](#)
- [AlertsRequestV1\\$isValid\(\)](#)
- [AlertsRequestV1\\$getInvalidFields\(\)](#)
- [AlertsRequestV1\\$print\(\)](#)
- [AlertsRequestV1\\$clone\(\)](#)

[AlertsRequestV1\\$new\(\)](#): Initialize a new AlertsRequestV1 class.

Usage:

```
AlertsRequestV1$new(
  context_enriched,
  metadata,
  date_from = NULL,
  date_to = NULL,
  filters = NULL,
  ...
)
```

Arguments:

`context_enriched` When true, treat the supplied metadata as already context-enriched.

`metadata` metadata

`date_from` Optional start date bound for alert detection (YYYY-MM-DD).

`date_to` Optional end date bound for alert detection (YYYY-MM-DD).

`filters` Optional. ‘limit’ controls the number of alerts returned (**0–1000**, default **100**). ‘categories[]’ and ‘regions[]’ narrow the alert universe; each must be an integer **1–9999**.

Values are not verified against catalog APIs.

... Other optional arguments.

AlertsRequestV1\$list(): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
AlertsRequestV1$list()
```

Returns: AlertsRequestV1 as a base R list.

Examples:

```
# convert array of AlertsRequestV1 (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$list()) |> map(as_tibble) |> list_rbind()
df
```

AlertsRequestV1\$toSimpleType(): Convert AlertsRequestV1 to a base R type

Usage:

```
AlertsRequestV1$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

AlertsRequestV1\$fromJSON(): Deserialize JSON string into an instance of AlertsRequestV1

Usage:

```
AlertsRequestV1$fromJSON(input_json)
```

Arguments:

`input_json` the JSON input

Returns: the instance of AlertsRequestV1

AlertsRequestV1\$toJSONString(): To JSON String

Usage:

AlertsRequestV1\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: AlertsRequestV1 in JSON format

AlertsRequestV1\$fromJSONString(): Deserialize JSON string into an instance of AlertsRequestV1

Usage:

AlertsRequestV1\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of AlertsRequestV1

AlertsRequestV1\$validateJSON(): Validate JSON input with respect to AlertsRequestV1 and throw an exception if invalid

Usage:

AlertsRequestV1\$validateJSON(input)

Arguments:

input the JSON input

AlertsRequestV1\$toString(): To string (JSON format)

Usage:

AlertsRequestV1\$toString()

Returns: String representation of AlertsRequestV1

AlertsRequestV1\$isValid(): Return true if the values in all fields are valid.

Usage:

AlertsRequestV1\$isValid()

Returns: true if the values in all fields are valid.

AlertsRequestV1\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

AlertsRequestV1\$getInvalidFields()

Returns: A list of invalid fields (if any).

AlertsRequestV1\$print(): Print the object

Usage:

AlertsRequestV1\$print()

AlertsRequestV1\$clone(): The objects of this class are cloneable with this method.

Usage:

AlertsRequestV1\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `AlertsRequestV1$toList()`
## -----

# convert array of AlertsRequestV1 (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

AnyType

AnyType base class for OpenAPI free-form objects.

Description

Base class used by models that may carry additional properties beyond their declared fields (OpenAPI ‘anyType’ / free-form schemas).

Methods

Public methods:

- [AnyType\\$print\(\)](#)
- [AnyType\\$toJSONString\(\)](#)
- [AnyType\\$toSimpleType\(\)](#)
- [AnyType\\$toList\(\)](#)
- [AnyType\\$fromJSON\(\)](#)
- [AnyType\\$fromJSONString\(\)](#)
- [AnyType\\$clone\(\)](#)

AnyType\$print(): Print the object as JSON.

Usage:

```
AnyType$print()
```

AnyType\$toJSONString(): Serialize to a JSON string.

Usage:

```
AnyType$toJSONString(...)
```

AnyType\$toSimpleType(): Convert to a base R type.

Usage:

```
AnyType$toSimpleType()
```

AnyType\$toList(): Convert to list (alias for toSimpleType).

Usage:

AnyType\$toList()

AnyType\$fromJSON(): Deserialize from a JSON string.

Usage:

AnyType\$fromJSON(input_json)

AnyType\$fromJSONString(): Deserialize from a JSON string (alias).

Usage:

AnyType\$fromJSONString(input_json)

AnyType\$clone(): The objects of this class are cloneable with this method.

Usage:

AnyType\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

ApiClient

ApiClient

Description

ApiClient Class

Format

An R6Class generator object

Details

Sybilion API

The Sybilion API powers the Sybilion Developers Portal: forecasts, drivers, catalog, account and usage. Authenticate every request with 'Authorization: Bearer <token>' using either an API key created in the Developers Portal or an Auth0 access token from your dashboard session.

The version of the OpenAPI document: 0.1.0 Generated by: <https://openapi-generator.tech>

ApiClient Class

Generic API client for OpenAPI client library builds. OpenAPI generic API client. This client handles the client- server communication, and is invariant across implementations. Specifics of the methods and models for each application are generated from the OpenAPI Generator templates.

NOTE: This class is auto generated by OpenAPI Generator (<https://openapi-generator.tech>). Ref: <https://openapi-generator.tech> Do not edit the class manually.

Public fields

base_path Base url
user_agent Default user agent
default_headers Default headers
username Username for HTTP basic authentication
password Password for HTTP basic authentication
api_keys API keys
bearer_token Bearer token
timeout Default timeout in seconds
retry_status_codes vector of status codes to retry
max_retry_attempts maximum number of retries for the status codes

Methods**Public methods:**

- [ApiClient\\$new\(\)](#)
- [ApiClient\\$CallApi\(\)](#)
- [ApiClient\\$Execute\(\)](#)
- [ApiClient\\$deserialize\(\)](#)
- [ApiClient\\$deserializeObj\(\)](#)
- [ApiClient\\$select_header\(\)](#)
- [ApiClient\\$DeserializeResponse\(\)](#)
- [ApiClient\\$WriteFile\(\)](#)
- [ApiClient\\$IsBinary\(\)](#)
- [ApiClient\\$clone\(\)](#)

`ApiClient$new()`: Initialize a new ApiClient.

Usage:

```
ApiClient$new(  
  base_path = NULL,  
  user_agent = NULL,  
  default_headers = NULL,  
  username = NULL,  
  password = NULL,  
  api_keys = NULL,  
  access_token = NULL,  
  bearer_token = NULL,  
  timeout = NULL,  
  retry_status_codes = NULL,  
  max_retry_attempts = NULL  
)
```

Arguments:

base_path Base path.

user_agent User agent.
 default_headers Default headers.
 username User name.
 password Password.
 api_keys API keys.
 access_token Access token.
 bearer_token Bearer token.
 timeout Timeout.
 retry_status_codes Status codes for retry.
 max_retry_attempts Maximum number of retry.

ApiClient\$CallApi(): Prepare to make an API call with the retry logic.

Usage:

```

ApiClient$CallApi(
    url,
    method,
    query_params,
    header_params,
    form_params,
    file_params,
    accepts,
    content_types,
    body,
    stream_callback = NULL,
    ...
)
  
```

Arguments:

url URL.
 method HTTP method.
 query_params The query parameters.
 header_params The header parameters.
 form_params The form parameters.
 file_params The form parameters for uploading files.
 accepts The list of Accept headers.
 content_types The list of Content-Type headers.
 body The HTTP request body.
 stream_callback Callback function to process the data stream
 ... Other optional arguments.

Returns: HTTP response

ApiClient\$Execute(): Make an API call

Usage:

```

ApiClient$Execute(
  url,
  method,
  query_params,
  header_params,
  form_params,
  file_params,
  accepts,
  content_types,
  body,
  stream_callback = NULL,
  ...
)

```

Arguments:

url URL.
 method HTTP method.
 query_params The query parameters.
 header_params The header parameters.
 form_params The form parameters.
 file_params The form parameters for uploading files.
 accepts The list of Accept headers
 content_types The list of Content-Type headers
 body The HTTP request body.
 stream_callback Callback function to process data stream
 ... Other optional arguments.

Returns: HTTP response

ApiClient\$deserialize(): Deserialize the content of API response to the given type.

Usage:

```
ApiClient$deserialize(raw_response, return_type, pkg_env)
```

Arguments:

raw_response Raw response.
 return_type R return type.
 pkg_env Package environment.

Returns: Deserialized object.

ApiClient\$deserializeObj(): Deserialize the response from jsonlite object based on the given type by handling complex and nested types by iterating recursively Example return_types will be like "array[integer]", "map(Pet)", "array[map(Tag)]", etc.,

Usage:

```
ApiClient$deserializeObj(obj, return_type, pkg_env)
```

Arguments:

obj Response object.

return_type R return type.
 pkg_env Package environment.
Returns: Deserialized object.

`ApiClient$select_header()`: Return a property header (for accept or content-type). If JSON-related MIME is found, return it. Otherwise, return the first one, if any.

Usage:
`ApiClient$select_header(headers)`
Arguments:
 headers A list of headers
Returns: A header (e.g. 'application/json')

`ApiClient$DeserializeResponse()`: Deserialize the response

Usage:
`ApiClient$DeserializeResponse(local_var_resp, return_type = NULL)`
Arguments:
 local_var_resp The API response
 return_type The target return type for the endpoint (e.g., "object"). If 'NULL' text will be left as-is.
Returns: If the raw response is coercable to text, return the text. Otherwise return the raw response.

`ApiClient$WriteFile()`: Write response to a file
 The function will write out data.

1. If binary data is detected it will use 'writeBin' 2. If the raw response is coercible to text, the text will be written to a file 3. If the raw response is not coercible to text, the raw response will be written

Usage:
`ApiClient$WriteFile(local_var_resp, file)`
Arguments:
 local_var_resp The API response
 file The name of the data file to save the result

`ApiClient$IsBinary()`: Check response for binary content

Usage:
`ApiClient$IsBinary(x)`
Arguments:
 local_var_resp The API response

`ApiClient$clone()`: The objects of this class are cloneable with this method.

Usage:
`ApiClient$clone(deep = FALSE)`
Arguments:
 deep Whether to make a deep clone.

ApiResponse

ApiResponse

Description

ApiResponse Class

Format

An R6Class generator object

Details

Sybillion API

The Sybillion API powers the Sybillion Developers Portal: forecasts, drivers, catalog, account and usage. Authenticate every request with 'Authorization: Bearer <token>' using either an API key created in the Developers Portal or an Auth0 access token from your dashboard session.

The version of the OpenAPI document: 0.1.0 Generated by: <https://openapi-generator.tech>

Public fields

content The deserialized response body.

response The raw response from the endpoint.

status_code The HTTP response status code.

status_code_desc The brief description of the HTTP response status code.

headers The HTTP response headers.

Methods

Public methods:

- [ApiResponse\\$new\(\)](#)
- [ApiResponse\\$response_as_text\(\)](#)
- [ApiResponse\\$clone\(\)](#)

`ApiResponse$new()`: Initialize a new ApiResponse class.

Usage:

```
ApiResponse$new(  
  content = NULL,  
  response = NULL,  
  status_code = NULL,  
  status_code_desc = NULL,  
  headers = NULL  
)
```

Arguments:

content The deserialized response body.
 response The raw response from the endpoint.
 status_code The HTTP response status code.
 status_code_desc The brief description of the HTTP response status code.
 headers The HTTP response headers.

`ApiResponse$response_as_text()`: The response is stored as a raw vector. Use this to access the response after converting it to text. If the response is not coercible to text NA is returned.

Usage:

```
ApiResponse$response_as_text(from_encoding = "", to_encoding = "UTF-8")
```

Arguments:

from_encoding The encoding of the raw response.
 to_encoding The target encoding of the return value.

`ApiResponse$clone()`: The objects of this class are cloneable with this method.

Usage:

```
ApiResponse$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

ApiV1AlertsPost200Response

ApiV1AlertsPost200Response

Description

ApiV1AlertsPost200Response Class

ApiV1AlertsPost200Response Class

Format

An R6Class generator object

Details

Create a new ApiV1AlertsPost200Response

Public fields

alerts list([AlertItemV1](#)) [optional]

Methods**Public methods:**

- `ApiV1AlertsPost200Response$new()`
- `ApiV1AlertsPost200Response$toList()`
- `ApiV1AlertsPost200Response$toSimpleType()`
- `ApiV1AlertsPost200Response$fromJSON()`
- `ApiV1AlertsPost200Response$toJSONString()`
- `ApiV1AlertsPost200Response$fromJSONString()`
- `ApiV1AlertsPost200Response$validateJSON()`
- `ApiV1AlertsPost200Response$toString()`
- `ApiV1AlertsPost200Response$isValid()`
- `ApiV1AlertsPost200Response$getInvalidFields()`
- `ApiV1AlertsPost200Response$print()`
- `ApiV1AlertsPost200Response$clone()`

`ApiV1AlertsPost200Response$new()`: Initialize a new `ApiV1AlertsPost200Response` class.

Usage:

```
ApiV1AlertsPost200Response$new(alerts = NULL, ...)
```

Arguments:

`alerts` alerts

... Other optional arguments.

`ApiV1AlertsPost200Response$toList()`: Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
ApiV1AlertsPost200Response$toList()
```

Returns: `ApiV1AlertsPost200Response` as a base R list.

Examples:

```
# convert array of ApiV1AlertsPost200Response (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

`ApiV1AlertsPost200Response$toSimpleType()`: Convert `ApiV1AlertsPost200Response` to a base R type

Usage:

```
ApiV1AlertsPost200Response$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

`ApiV1AlertsPost200Response$fromJSON()`: Deserialize JSON string into an instance of `ApiV1AlertsPost200Response`

Usage:

ApiV1AlertsPost200Response\$fromJSON(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ApiV1AlertsPost200Response

ApiV1AlertsPost200Response\$toJSONString(): To JSON String

Usage:

ApiV1AlertsPost200Response\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: ApiV1AlertsPost200Response in JSON format

ApiV1AlertsPost200Response\$fromJSONString(): Deserialize JSON string into an instance of ApiV1AlertsPost200Response

Usage:

ApiV1AlertsPost200Response\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ApiV1AlertsPost200Response

ApiV1AlertsPost200Response\$validateJSON(): Validate JSON input with respect to ApiV1AlertsPost200Response and throw an exception if invalid

Usage:

ApiV1AlertsPost200Response\$validateJSON(input)

Arguments:

input the JSON input

ApiV1AlertsPost200Response\$toString(): To string (JSON format)

Usage:

ApiV1AlertsPost200Response\$toString()

Returns: String representation of ApiV1AlertsPost200Response

ApiV1AlertsPost200Response\$isValid(): Return true if the values in all fields are valid.

Usage:

ApiV1AlertsPost200Response\$isValid()

Returns: true if the values in all fields are valid.

ApiV1AlertsPost200Response\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

ApiV1AlertsPost200Response\$getInvalidFields()

Returns: A list of invalid fields (if any).

ApiV1AlertsPost200Response\$print(): Print the object

Usage:

```
ApiV1AlertsPost200Response$print()
```

ApiV1AlertsPost200Response\$clone(): The objects of this class are cloneable with this method.

Usage:

```
ApiV1AlertsPost200Response$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `ApiV1AlertsPost200Response$toList()`
## -----

# convert array of ApiV1AlertsPost200Response (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

ApiV1DriversPost200Response
ApiV1DriversPost200Response

Description

ApiV1DriversPost200Response Class
 ApiV1DriversPost200Response Class

Format

An R6Class generator object

Details

Create a new ApiV1DriversPost200Response

Public fields

data [ApiV1DriversPost200ResponseData](#) [optional]
 message character [optional]
 status integer [optional]

Methods

Public methods:

- `ApiV1DriversPost200Response$new()`
- `ApiV1DriversPost200Response$toList()`
- `ApiV1DriversPost200Response$toSimpleType()`
- `ApiV1DriversPost200Response$fromJSON()`
- `ApiV1DriversPost200Response$toJSONString()`
- `ApiV1DriversPost200Response$fromJSONString()`
- `ApiV1DriversPost200Response$validateJSON()`
- `ApiV1DriversPost200Response$toString()`
- `ApiV1DriversPost200Response$isValid()`
- `ApiV1DriversPost200Response$getInvalidFields()`
- `ApiV1DriversPost200Response$print()`
- `ApiV1DriversPost200Response$clone()`

`ApiV1DriversPost200Response$new()`: Initialize a new `ApiV1DriversPost200Response` class.

Usage:

```
ApiV1DriversPost200Response$new(
  data = NULL,
  message = NULL,
  status = NULL,
  ...
)
```

Arguments:

```
data data
message message
status status
... Other optional arguments.
```

`ApiV1DriversPost200Response$toList()`: Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
ApiV1DriversPost200Response$toList()
```

Returns: `ApiV1DriversPost200Response` as a base R list.

Examples:

```
# convert array of ApiV1DriversPost200Response (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

`ApiV1DriversPost200Response$toSimpleType()`: Convert `ApiV1DriversPost200Response` to a base R type

Usage:

ApiV1DriversPost200Response\$toSimpleType()

Returns: A base R type, e.g. a list or numeric/character array.

ApiV1DriversPost200Response\$fromJSON(): Deserialize JSON string into an instance of ApiV1DriversPost200Response

Usage:

ApiV1DriversPost200Response\$fromJSON(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ApiV1DriversPost200Response

ApiV1DriversPost200Response\$toJSONString(): To JSON String

Usage:

ApiV1DriversPost200Response\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: ApiV1DriversPost200Response in JSON format

ApiV1DriversPost200Response\$fromJSONString(): Deserialize JSON string into an instance of ApiV1DriversPost200Response

Usage:

ApiV1DriversPost200Response\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ApiV1DriversPost200Response

ApiV1DriversPost200Response\$validateJSON(): Validate JSON input with respect to ApiV1DriversPost200Response and throw an exception if invalid

Usage:

ApiV1DriversPost200Response\$validateJSON(input)

Arguments:

input the JSON input

ApiV1DriversPost200Response\$toString(): To string (JSON format)

Usage:

ApiV1DriversPost200Response\$toString()

Returns: String representation of ApiV1DriversPost200Response

ApiV1DriversPost200Response\$isValid(): Return true if the values in all fields are valid.

Usage:

ApiV1DriversPost200Response\$isValid()

Returns: true if the values in all fields are valid.

ApiV1DriversPost200Response\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

```
ApiV1DriversPost200Response$getInvalidFields()
```

Returns: A list of invalid fields (if any).

ApiV1DriversPost200Response\$print(): Print the object

Usage:

```
ApiV1DriversPost200Response$print()
```

ApiV1DriversPost200Response\$clone(): The objects of this class are cloneable with this method.

Usage:

```
ApiV1DriversPost200Response$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `ApiV1DriversPost200Response$toList()`
## -----

# convert array of ApiV1DriversPost200Response (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

ApiV1DriversPost200ResponseData

ApiV1DriversPost200ResponseData

Description

ApiV1DriversPost200ResponseData Class

ApiV1DriversPost200ResponseData Class

Format

An R6Class generator object

Details

Create a new ApiV1DriversPost200ResponseData

Public fields

drivers list([DriverItemV1](#)) [optional]

Methods**Public methods:**

- [ApiV1DriversPost200ResponseData\\$new\(\)](#)
- [ApiV1DriversPost200ResponseData\\$toList\(\)](#)
- [ApiV1DriversPost200ResponseData\\$toSimpleType\(\)](#)
- [ApiV1DriversPost200ResponseData\\$fromJSON\(\)](#)
- [ApiV1DriversPost200ResponseData\\$toJSONString\(\)](#)
- [ApiV1DriversPost200ResponseData\\$fromJSONString\(\)](#)
- [ApiV1DriversPost200ResponseData\\$validateJSON\(\)](#)
- [ApiV1DriversPost200ResponseData\\$toString\(\)](#)
- [ApiV1DriversPost200ResponseData\\$isValid\(\)](#)
- [ApiV1DriversPost200ResponseData\\$getInvalidFields\(\)](#)
- [ApiV1DriversPost200ResponseData\\$print\(\)](#)
- [ApiV1DriversPost200ResponseData\\$clone\(\)](#)

[ApiV1DriversPost200ResponseData\\$new\(\)](#): Initialize a new ApiV1DriversPost200ResponseData class.

Usage:

```
ApiV1DriversPost200ResponseData$new(drivers = NULL, ...)
```

Arguments:

drivers drivers

... Other optional arguments.

[ApiV1DriversPost200ResponseData\\$toList\(\)](#): Convert to a List
Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
ApiV1DriversPost200ResponseData$toList()
```

Returns: ApiV1DriversPost200ResponseData as a base R list.

Examples:

```
# convert array of ApiV1DriversPost200ResponseData (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

[ApiV1DriversPost200ResponseData\\$toSimpleType\(\)](#): Convert ApiV1DriversPost200ResponseData to a base R type

Usage:

ApiV1DriversPost200ResponseData\$toSimpleType()

Returns: A base R type, e.g. a list or numeric/character array.

ApiV1DriversPost200ResponseData\$fromJSON(): Deserialize JSON string into an instance of ApiV1DriversPost200ResponseData

Usage:

ApiV1DriversPost200ResponseData\$fromJSON(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ApiV1DriversPost200ResponseData

ApiV1DriversPost200ResponseData\$toJSONString(): To JSON String

Usage:

ApiV1DriversPost200ResponseData\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: ApiV1DriversPost200ResponseData in JSON format

ApiV1DriversPost200ResponseData\$fromJSONString(): Deserialize JSON string into an instance of ApiV1DriversPost200ResponseData

Usage:

ApiV1DriversPost200ResponseData\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ApiV1DriversPost200ResponseData

ApiV1DriversPost200ResponseData\$validateJSON(): Validate JSON input with respect to ApiV1DriversPost200ResponseData and throw an exception if invalid

Usage:

ApiV1DriversPost200ResponseData\$validateJSON(input)

Arguments:

input the JSON input

ApiV1DriversPost200ResponseData\$toString(): To string (JSON format)

Usage:

ApiV1DriversPost200ResponseData\$toString()

Returns: String representation of ApiV1DriversPost200ResponseData

ApiV1DriversPost200ResponseData\$isValid(): Return true if the values in all fields are valid.

Usage:

ApiV1DriversPost200ResponseData\$isValid()

Returns: true if the values in all fields are valid.

ApiV1DriversPost200ResponseData\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

ApiV1DriversPost200ResponseData\$getInvalidFields()

Returns: A list of invalid fields (if any).

ApiV1DriversPost200ResponseData\$print(): Print the object

Usage:

ApiV1DriversPost200ResponseData\$print()

ApiV1DriversPost200ResponseData\$clone(): The objects of this class are cloneable with this method.

Usage:

ApiV1DriversPost200ResponseData\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `ApiV1DriversPost200ResponseData$toList()`
## -----

# convert array of ApiV1DriversPost200ResponseData (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

ApiV1ForecastsIdGet200Response

ApiV1ForecastsIdGet200Response

Description

ApiV1ForecastsIdGet200Response Class

ApiV1ForecastsIdGet200Response Class

Format

An R6Class generator object

Details

Create a new ApiV1ForecastsIdGet200Response

Public fields

artifacts List of output files available for download. Populated once the job completes. list([ForecastArtifactMeta](#)) [optional]

job_id character [optional]

pipeline_error Structured error from the pipeline for failed or canceled jobs; null otherwise. Common fields: 'code' (string error code) and 'detail' (human-readable explanation). Shape is defined by the pipeline and may vary. object [optional]

settled True once the job has reached a terminal state and the charge has been posted. character [optional]

status Current lifecycle state of the job. character [optional]

Methods**Public methods:**

- [ApiV1ForecastsIdGet200Response\\$new\(\)](#)
- [ApiV1ForecastsIdGet200Response\\$toList\(\)](#)
- [ApiV1ForecastsIdGet200Response\\$toSimpleType\(\)](#)
- [ApiV1ForecastsIdGet200Response\\$fromJSON\(\)](#)
- [ApiV1ForecastsIdGet200Response\\$toJSONString\(\)](#)
- [ApiV1ForecastsIdGet200Response\\$fromJSONString\(\)](#)
- [ApiV1ForecastsIdGet200Response\\$validateJSON\(\)](#)
- [ApiV1ForecastsIdGet200Response\\$toString\(\)](#)
- [ApiV1ForecastsIdGet200Response\\$isValid\(\)](#)
- [ApiV1ForecastsIdGet200Response\\$getInvalidFields\(\)](#)
- [ApiV1ForecastsIdGet200Response\\$print\(\)](#)
- [ApiV1ForecastsIdGet200Response\\$clone\(\)](#)

[ApiV1ForecastsIdGet200Response\\$new\(\)](#): Initialize a new ApiV1ForecastsIdGet200Response class.

Usage:

```
ApiV1ForecastsIdGet200Response$new(
  artifacts = NULL,
  job_id = NULL,
  pipeline_error = NULL,
  settled = NULL,
  status = NULL,
  ...
)
```

Arguments:

`artifacts` List of output files available for download. Populated once the job completes.
`job_id` `job_id`
`pipeline_error` Structured error from the pipeline for failed or canceled jobs; null otherwise.
 Common fields: 'code' (string error code) and 'detail' (human-readable explanation). Shape is defined by the pipeline and may vary.
`settled` True once the job has reached a terminal state and the charge has been posted.
`status` Current lifecycle state of the job.
 ... Other optional arguments.

`ApiV1ForecastsIdGet200Response$toList()`: Convert to a List
 Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
ApiV1ForecastsIdGet200Response$toList()
```

Returns: `ApiV1ForecastsIdGet200Response` as a base R list.

Examples:

```
# convert array of ApiV1ForecastsIdGet200Response (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

`ApiV1ForecastsIdGet200Response$toSimpleType()`: Convert `ApiV1ForecastsIdGet200Response` to a base R type

Usage:

```
ApiV1ForecastsIdGet200Response$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

`ApiV1ForecastsIdGet200Response$fromJSON()`: Deserialize JSON string into an instance of `ApiV1ForecastsIdGet200Response`

Usage:

```
ApiV1ForecastsIdGet200Response$fromJSON(input_json)
```

Arguments:

`input_json` the JSON input

Returns: the instance of `ApiV1ForecastsIdGet200Response`

`ApiV1ForecastsIdGet200Response$toJSONString()`: To JSON String

Usage:

```
ApiV1ForecastsIdGet200Response$toJSONString(...)
```

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: `ApiV1ForecastsIdGet200Response` in JSON format

ApiV1ForecastsIdGet200Response\$fromJSONString(): Deserialize JSON string into an instance of `ApiV1ForecastsIdGet200Response`

Usage:

`ApiV1ForecastsIdGet200Response$fromJSONString(input_json)`

Arguments:

`input_json` the JSON input

Returns: the instance of `ApiV1ForecastsIdGet200Response`

ApiV1ForecastsIdGet200Response\$validateJSON(): Validate JSON input with respect to `ApiV1ForecastsIdGet200Response` and throw an exception if invalid

Usage:

`ApiV1ForecastsIdGet200Response$validateJSON(input)`

Arguments:

`input` the JSON input

ApiV1ForecastsIdGet200Response\$toString(): To string (JSON format)

Usage:

`ApiV1ForecastsIdGet200Response$toString()`

Returns: String representation of `ApiV1ForecastsIdGet200Response`

ApiV1ForecastsIdGet200Response\$isValid(): Return true if the values in all fields are valid.

Usage:

`ApiV1ForecastsIdGet200Response$isValid()`

Returns: true if the values in all fields are valid.

ApiV1ForecastsIdGet200Response\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

`ApiV1ForecastsIdGet200Response$getInvalidFields()`

Returns: A list of invalid fields (if any).

ApiV1ForecastsIdGet200Response\$print(): Print the object

Usage:

`ApiV1ForecastsIdGet200Response$print()`

ApiV1ForecastsIdGet200Response\$clone(): The objects of this class are cloneable with this method.

Usage:

`ApiV1ForecastsIdGet200Response$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Examples

```
## -----
## Method `ApiV1ForecastsIdGet200Response$toList()`
## -----

# convert array of ApiV1ForecastsIdGet200Response (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

ApiV1ForecastsPost202Response

ApiV1ForecastsPost202Response

Description

ApiV1ForecastsPost202Response Class

ApiV1ForecastsPost202Response Class

Format

An R6Class generator object

Details

Create a new ApiV1ForecastsPost202Response

Public fields

job_id Unique job id — use this to poll status and download artifacts. character [optional]

poll_url Convenience URL for polling this job's status. character [optional]

run_id Internal run identifier (opaque; useful for support). character [optional]

workflow Internal workflow identifier (opaque; useful for support). character [optional]

Methods

Public methods:

- [ApiV1ForecastsPost202Response\\$new\(\)](#)
- [ApiV1ForecastsPost202Response\\$toList\(\)](#)
- [ApiV1ForecastsPost202Response\\$toSimpleType\(\)](#)
- [ApiV1ForecastsPost202Response\\$fromJSON\(\)](#)
- [ApiV1ForecastsPost202Response\\$toJSONString\(\)](#)

- `ApiV1ForecastsPost202Response$fromJSONString()`
- `ApiV1ForecastsPost202Response$validateJSON()`
- `ApiV1ForecastsPost202Response$toString()`
- `ApiV1ForecastsPost202Response$isValid()`
- `ApiV1ForecastsPost202Response$getInvalidFields()`
- `ApiV1ForecastsPost202Response$print()`
- `ApiV1ForecastsPost202Response$clone()`

`ApiV1ForecastsPost202Response$new()`: Initialize a new `ApiV1ForecastsPost202Response` class.

Usage:

```
ApiV1ForecastsPost202Response$new(
  job_id = NULL,
  poll_url = NULL,
  run_id = NULL,
  workflow = NULL,
  ...
)
```

Arguments:

`job_id` Unique job id — use this to poll status and download artifacts.
`poll_url` Convenience URL for polling this job's status.
`run_id` Internal run identifier (opaque; useful for support).
`workflow` Internal workflow identifier (opaque; useful for support).
`...` Other optional arguments.

`ApiV1ForecastsPost202Response$toList()`: Convert to a List
 Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
ApiV1ForecastsPost202Response$toList()
```

Returns: `ApiV1ForecastsPost202Response` as a base R list.

Examples:

```
# convert array of ApiV1ForecastsPost202Response (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

`ApiV1ForecastsPost202Response$toSimpleType()`: Convert `ApiV1ForecastsPost202Response` to a base R type

Usage:

```
ApiV1ForecastsPost202Response$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

`ApiV1ForecastsPost202Response$fromJSON()`: Deserialize JSON string into an instance of `ApiV1ForecastsPost202Response`

Usage:

ApiV1ForecastsPost202Response\$fromJSON(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ApiV1ForecastsPost202Response

ApiV1ForecastsPost202Response\$toJSONString(): To JSON String

Usage:

ApiV1ForecastsPost202Response\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: ApiV1ForecastsPost202Response in JSON format

ApiV1ForecastsPost202Response\$fromJSONString(): Deserialize JSON string into an instance of ApiV1ForecastsPost202Response

Usage:

ApiV1ForecastsPost202Response\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ApiV1ForecastsPost202Response

ApiV1ForecastsPost202Response\$validateJSON(): Validate JSON input with respect to ApiV1ForecastsPost202Response and throw an exception if invalid

Usage:

ApiV1ForecastsPost202Response\$validateJSON(input)

Arguments:

input the JSON input

ApiV1ForecastsPost202Response\$toString(): To string (JSON format)

Usage:

ApiV1ForecastsPost202Response\$toString()

Returns: String representation of ApiV1ForecastsPost202Response

ApiV1ForecastsPost202Response\$isValid(): Return true if the values in all fields are valid.

Usage:

ApiV1ForecastsPost202Response\$isValid()

Returns: true if the values in all fields are valid.

ApiV1ForecastsPost202Response\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

ApiV1ForecastsPost202Response\$getInvalidFields()

Returns: A list of invalid fields (if any).

ApiV1ForecastsPost202Response\$print(): Print the object

Usage:

ApiV1ForecastsPost202Response\$print()

ApiV1ForecastsPost202Response\$clone(): The objects of this class are cloneable with this method.

Usage:

ApiV1ForecastsPost202Response\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `ApiV1ForecastsPost202Response$toList()`
## -----

# convert array of ApiV1ForecastsPost202Response (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

ApiV1JobsGet200Response

ApiV1JobsGet200Response

Description

ApiV1JobsGet200Response Class

ApiV1JobsGet200Response Class

Format

An R6Class generator object

Details

Create a new ApiV1JobsGet200Response

Public fields

jobs list([JobSummary](#))
 pagination [JobsPagination](#)

Methods**Public methods:**

- [ApiV1JobsGet200Response\\$new\(\)](#)
- [ApiV1JobsGet200Response\\$toList\(\)](#)
- [ApiV1JobsGet200Response\\$toSimpleType\(\)](#)
- [ApiV1JobsGet200Response\\$fromJSON\(\)](#)
- [ApiV1JobsGet200Response\\$toJSONString\(\)](#)
- [ApiV1JobsGet200Response\\$fromJSONString\(\)](#)
- [ApiV1JobsGet200Response\\$validateJSON\(\)](#)
- [ApiV1JobsGet200Response\\$toString\(\)](#)
- [ApiV1JobsGet200Response\\$isValid\(\)](#)
- [ApiV1JobsGet200Response\\$getInvalidFields\(\)](#)
- [ApiV1JobsGet200Response\\$print\(\)](#)
- [ApiV1JobsGet200Response\\$clone\(\)](#)

[ApiV1JobsGet200Response\\$new\(\)](#): Initialize a new [ApiV1JobsGet200Response](#) class.

Usage:

```
ApiV1JobsGet200Response$new(jobs, pagination, ...)
```

Arguments:

jobs jobs
 pagination pagination
 ... Other optional arguments.

[ApiV1JobsGet200Response\\$toList\(\)](#): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
ApiV1JobsGet200Response$toList()
```

Returns: [ApiV1JobsGet200Response](#) as a base R list.

Examples:

```
# convert array of ApiV1JobsGet200Response (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

[ApiV1JobsGet200Response\\$toSimpleType\(\)](#): Convert [ApiV1JobsGet200Response](#) to a base R type

Usage:

ApiV1JobsGet200Response\$toSimpleType()

Returns: A base R type, e.g. a list or numeric/character array.

ApiV1JobsGet200Response\$fromJSON(): Deserialize JSON string into an instance of ApiV1JobsGet200Response

Usage:

ApiV1JobsGet200Response\$fromJSON(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ApiV1JobsGet200Response

ApiV1JobsGet200Response\$toJSONString(): To JSON String

Usage:

ApiV1JobsGet200Response\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: ApiV1JobsGet200Response in JSON format

ApiV1JobsGet200Response\$fromJSONString(): Deserialize JSON string into an instance of ApiV1JobsGet200Response

Usage:

ApiV1JobsGet200Response\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ApiV1JobsGet200Response

ApiV1JobsGet200Response\$validateJSON(): Validate JSON input with respect to ApiV1JobsGet200Response and throw an exception if invalid

Usage:

ApiV1JobsGet200Response\$validateJSON(input)

Arguments:

input the JSON input

ApiV1JobsGet200Response\$toString(): To string (JSON format)

Usage:

ApiV1JobsGet200Response\$toString()

Returns: String representation of ApiV1JobsGet200Response

ApiV1JobsGet200Response\$isValid(): Return true if the values in all fields are valid.

Usage:

ApiV1JobsGet200Response\$isValid()

Returns: true if the values in all fields are valid.

ApiV1JobsGet200Response\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

```
ApiV1JobsGet200Response$getInvalidFields()
```

Returns: A list of invalid fields (if any).

ApiV1JobsGet200Response\$print(): Print the object

Usage:

```
ApiV1JobsGet200Response$print()
```

ApiV1JobsGet200Response\$clone(): The objects of this class are cloneable with this method.

Usage:

```
ApiV1JobsGet200Response$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `ApiV1JobsGet200Response$toList()`
## -----

# convert array of ApiV1JobsGet200Response (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

ApiV1UsageGet200Response

ApiV1UsageGet200Response

Description

ApiV1UsageGet200Response Class

ApiV1UsageGet200Response Class

Format

An R6Class generator object

Details

Create a new ApiV1UsageGet200Response

Public fields

pagination [Pagination](#)
 usage_events list([UsageEvent](#))

Methods**Public methods:**

- [ApiV1UsageGet200Response\\$new\(\)](#)
- [ApiV1UsageGet200Response\\$toList\(\)](#)
- [ApiV1UsageGet200Response\\$toSimpleType\(\)](#)
- [ApiV1UsageGet200Response\\$fromJSON\(\)](#)
- [ApiV1UsageGet200Response\\$toJSONString\(\)](#)
- [ApiV1UsageGet200Response\\$fromJSONString\(\)](#)
- [ApiV1UsageGet200Response\\$validateJSON\(\)](#)
- [ApiV1UsageGet200Response\\$toString\(\)](#)
- [ApiV1UsageGet200Response\\$isValid\(\)](#)
- [ApiV1UsageGet200Response\\$getInvalidFields\(\)](#)
- [ApiV1UsageGet200Response\\$print\(\)](#)
- [ApiV1UsageGet200Response\\$clone\(\)](#)

[ApiV1UsageGet200Response\\$new\(\)](#): Initialize a new [ApiV1UsageGet200Response](#) class.

Usage:

```
ApiV1UsageGet200Response$new(pagination, usage_events, ...)
```

Arguments:

pagination pagination
 usage_events usage_events
 ... Other optional arguments.

[ApiV1UsageGet200Response\\$toList\(\)](#): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
ApiV1UsageGet200Response$toList()
```

Returns: [ApiV1UsageGet200Response](#) as a base R list.

Examples:

```
# convert array of ApiV1UsageGet200Response (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

[ApiV1UsageGet200Response\\$toSimpleType\(\)](#): Convert [ApiV1UsageGet200Response](#) to a base R type

Usage:

`ApiV1UsageGet200Response$toSimpleType()`

Returns: A base R type, e.g. a list or numeric/character array.

`ApiV1UsageGet200Response$fromJSON()`: Deserialize JSON string into an instance of `ApiV1UsageGet200Response`

Usage:

`ApiV1UsageGet200Response$fromJSON(input_json)`

Arguments:

`input_json` the JSON input

Returns: the instance of `ApiV1UsageGet200Response`

`ApiV1UsageGet200Response$toJSONString()`: To JSON String

Usage:

`ApiV1UsageGet200Response$toJSONString(...)`

Arguments:

... Parameters passed to `'jsonlite::toJSON'`

Returns: `ApiV1UsageGet200Response` in JSON format

`ApiV1UsageGet200Response$fromJSONString()`: Deserialize JSON string into an instance of `ApiV1UsageGet200Response`

Usage:

`ApiV1UsageGet200Response$fromJSONString(input_json)`

Arguments:

`input_json` the JSON input

Returns: the instance of `ApiV1UsageGet200Response`

`ApiV1UsageGet200Response$validateJSON()`: Validate JSON input with respect to `ApiV1UsageGet200Response` and throw an exception if invalid

Usage:

`ApiV1UsageGet200Response$validateJSON(input)`

Arguments:

`input` the JSON input

`ApiV1UsageGet200Response$toString()`: To string (JSON format)

Usage:

`ApiV1UsageGet200Response$toString()`

Returns: String representation of `ApiV1UsageGet200Response`

`ApiV1UsageGet200Response$isValid()`: Return true if the values in all fields are valid.

Usage:

`ApiV1UsageGet200Response$isValid()`

Returns: true if the values in all fields are valid.

ApiV1UsageGet200Response\$getInvalidFields(): Return a list of invalid fields (if any).
Usage:
ApiV1UsageGet200Response\$getInvalidFields()
Returns: A list of invalid fields (if any).

ApiV1UsageGet200Response\$print(): Print the object
Usage:
ApiV1UsageGet200Response\$print()

ApiV1UsageGet200Response\$clone(): The objects of this class are cloneable with this method.
Usage:
ApiV1UsageGet200Response\$clone(deep = FALSE)
Arguments:
deep Whether to make a deep clone.

Examples

```
## -----  
## Method `ApiV1UsageGet200Response$toList()`  
## -----  
  
# convert array of ApiV1UsageGet200Response (x) to a data frame  
## Not run:  
library(purrr)  
library(tibble)  
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()  
df  
  
## End(Not run)
```

AutoRechargeState	<i>AutoRechargeState</i>
-------------------	--------------------------

Description

AutoRechargeState Class
AutoRechargeState Class

Format

An R6Class generator object

Details

Create a new AutoRechargeState

Public fields

`below_eur_cents` When the available balance drops below this many EUR cents, a recharge is triggered. integer

`enabled` Whether auto-recharge is active for this account. character

`has_stripe_customer` Whether a Stripe customer record exists (required for auto-recharge to run). character

`meter_cents` EUR cents charged via auto-recharge in the current UTC calendar month. integer

`meter_month` UTC month start for 'meter_cents'; null if no auto-recharge has run this month. character [optional]

`monthly_cap_cents` Maximum EUR cents that may be charged via auto-recharge per UTC calendar month. 0 = no cap. integer

`target_eur_cents` Balance target after a successful recharge, in EUR cents. integer

Methods**Public methods:**

- `AutoRechargeState$new()`
- `AutoRechargeState$toList()`
- `AutoRechargeState$toSimpleType()`
- `AutoRechargeState$fromJSON()`
- `AutoRechargeState$toJSONString()`
- `AutoRechargeState$fromJSONString()`
- `AutoRechargeState$validateJSON()`
- `AutoRechargeState$toString()`
- `AutoRechargeState$isValid()`
- `AutoRechargeState$getInvalidFields()`
- `AutoRechargeState$print()`
- `AutoRechargeState$clone()`

`AutoRechargeState$new()`: Initialize a new `AutoRechargeState` class.

Usage:

```
AutoRechargeState$new(
  below_eur_cents,
  enabled,
  has_stripe_customer,
  meter_cents,
  monthly_cap_cents,
  target_eur_cents,
  meter_month = NULL,
  ...
)
```

Arguments:

`below_eur_cents` When the available balance drops below this many EUR cents, a recharge is triggered.

enabled Whether auto-recharge is active for this account.
has_stripe_customer Whether a Stripe customer record exists (required for auto-recharge to run).
meter_cents EUR cents charged via auto-recharge in the current UTC calendar month.
monthly_cap_cents Maximum EUR cents that may be charged via auto-recharge per UTC calendar month. 0 = no cap.
target_eur_cents Balance target after a successful recharge, in EUR cents.
meter_month UTC month start for 'meter_cents'; null if no auto-recharge has run this month.
 ... Other optional arguments.

AutoRechargeState\$toList(): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
AutoRechargeState$toList()
```

Returns: AutoRechargeState as a base R list.

Examples:

```
# convert array of AutoRechargeState (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

AutoRechargeState\$toSimpleType(): Convert AutoRechargeState to a base R type

Usage:

```
AutoRechargeState$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

AutoRechargeState\$fromJSON(): Deserialize JSON string into an instance of AutoRechargeState

Usage:

```
AutoRechargeState$fromJSON(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of AutoRechargeState

AutoRechargeState\$toJSONString(): To JSON String

Usage:

```
AutoRechargeState$toJSONString(...)
```

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: AutoRechargeState in JSON format

AutoRechargeState\$fromJSONString(): Deserialize JSON string into an instance of AutoRechargeState

Usage:

AutoRechargeState\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of AutoRechargeState

AutoRechargeState\$validateJSON(): Validate JSON input with respect to AutoRechargeState and throw an exception if invalid

Usage:

AutoRechargeState\$validateJSON(input)

Arguments:

input the JSON input

AutoRechargeState\$toString(): To string (JSON format)

Usage:

AutoRechargeState\$toString()

Returns: String representation of AutoRechargeState

AutoRechargeState\$isValid(): Return true if the values in all fields are valid.

Usage:

AutoRechargeState\$isValid()

Returns: true if the values in all fields are valid.

AutoRechargeState\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

AutoRechargeState\$getInvalidFields()

Returns: A list of invalid fields (if any).

AutoRechargeState\$print(): Print the object

Usage:

AutoRechargeState\$print()

AutoRechargeState\$clone(): The objects of this class are cloneable with this method.

Usage:

AutoRechargeState\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `AutoRechargeState$toList()`
## -----

# convert array of AutoRechargeState (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

CategoryItemV1	<i>CategoryItemV1</i>
----------------	-----------------------

Description

A single thematic category returned by GET /api/v1/categories.
CategoryItemV1 Class

Format

An R6Class generator object

Details

Create a new CategoryItemV1

Public fields

id Integer identifier. Use this value in filters.categories[]. integer
name Human-readable category label. character

Methods

- Public methods:**
- [CategoryItemV1\\$new\(\)](#)
 - [CategoryItemV1\\$toList\(\)](#)
 - [CategoryItemV1\\$toSimpleType\(\)](#)
 - [CategoryItemV1\\$fromJSON\(\)](#)
 - [CategoryItemV1\\$toJSONString\(\)](#)
 - [CategoryItemV1\\$fromJSONString\(\)](#)
 - [CategoryItemV1\\$validateJSON\(\)](#)
 - [CategoryItemV1\\$toString\(\)](#)

- `CategoryItemV1$isValid()`
- `CategoryItemV1$getInvalidFields()`
- `CategoryItemV1$print()`
- `CategoryItemV1$clone()`

`CategoryItemV1$new()`: Initialize a new `CategoryItemV1` class.

Usage:

```
CategoryItemV1$new(id, name, ...)
```

Arguments:

`id` Integer identifier. Use this value in `filters.categories[]`.
`name` Human-readable category label.
`...` Other optional arguments.

`CategoryItemV1$list()`: Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
CategoryItemV1$list()
```

Returns: `CategoryItemV1` as a base R list.

Examples:

```
# convert array of CategoryItemV1 (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$list()) |> map(as_tibble) |> list_rbind()
df
```

`CategoryItemV1$toSimpleType()`: Convert `CategoryItemV1` to a base R type

Usage:

```
CategoryItemV1$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

`CategoryItemV1$fromJSON()`: Deserialize JSON string into an instance of `CategoryItemV1`

Usage:

```
CategoryItemV1$fromJSON(input_json)
```

Arguments:

`input_json` the JSON input

Returns: the instance of `CategoryItemV1`

`CategoryItemV1$toJSONString()`: To JSON String

Usage:

```
CategoryItemV1$toJSONString(...)
```

Arguments:

`...` Parameters passed to `'jsonlite::toJSON'`

Returns: CategoryItemV1 in JSON format

CategoryItemV1\$fromJsonString(): Deserialize JSON string into an instance of CategoryItemV1

Usage:

CategoryItemV1\$fromJsonString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of CategoryItemV1

CategoryItemV1\$validateJSON(): Validate JSON input with respect to CategoryItemV1 and throw an exception if invalid

Usage:

CategoryItemV1\$validateJSON(input)

Arguments:

input the JSON input

CategoryItemV1\$toString(): To string (JSON format)

Usage:

CategoryItemV1\$toString()

Returns: String representation of CategoryItemV1

CategoryItemV1\$isValid(): Return true if the values in all fields are valid.

Usage:

CategoryItemV1\$isValid()

Returns: true if the values in all fields are valid.

CategoryItemV1\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

CategoryItemV1\$getInvalidFields()

Returns: A list of invalid fields (if any).

CategoryItemV1\$print(): Print the object

Usage:

CategoryItemV1\$print()

CategoryItemV1\$clone(): The objects of this class are cloneable with this method.

Usage:

CategoryItemV1\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `CategoryItemV1$toList()`
## -----

# convert array of CategoryItemV1 (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

CategoryListResponse *CategoryListResponse*

Description

CategoryListResponse Class

CategoryListResponse Class

Format

An R6Class generator object

Details

Create a new CategoryListResponse

Public fields

items Complete category listing, sorted by id ascending. list([CategoryItemV1](#))

Methods**Public methods:**

- [CategoryListResponse\\$new\(\)](#)
- [CategoryListResponse\\$toList\(\)](#)
- [CategoryListResponse\\$toSimpleType\(\)](#)
- [CategoryListResponse\\$fromJSON\(\)](#)
- [CategoryListResponse\\$toJSONString\(\)](#)
- [CategoryListResponse\\$fromJSONString\(\)](#)
- [CategoryListResponse\\$validateJSON\(\)](#)
- [CategoryListResponse\\$toString\(\)](#)
- [CategoryListResponse\\$isValid\(\)](#)

- `CategoryListResponse$getInvalidFields()`
- `CategoryListResponse$print()`
- `CategoryListResponse$clone()`

`CategoryListResponse$new()`: Initialize a new `CategoryListResponse` class.

Usage:

```
CategoryListResponse$new(items, ...)
```

Arguments:

`items` Complete category listing, sorted by id ascending. No pagination.

`...` Other optional arguments.

`CategoryListResponse$toList()`: Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
CategoryListResponse$toList()
```

Returns: `CategoryListResponse` as a base R list.

Examples:

```
# convert array of CategoryListResponse (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

`CategoryListResponse$toSimpleType()`: Convert `CategoryListResponse` to a base R type

Usage:

```
CategoryListResponse$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

`CategoryListResponse$fromJSON()`: Deserialize JSON string into an instance of `CategoryListResponse`

Usage:

```
CategoryListResponse$fromJSON(input_json)
```

Arguments:

`input_json` the JSON input

Returns: the instance of `CategoryListResponse`

`CategoryListResponse$toJSONString()`: To JSON String

Usage:

```
CategoryListResponse$toJSONString(...)
```

Arguments:

`...` Parameters passed to `'jsonlite::toJSON'`

Returns: `CategoryListResponse` in JSON format

CategoryListResponse\$fromJSONString(): Deserialize JSON string into an instance of CategoryListResponse

Usage:

CategoryListResponse\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of CategoryListResponse

CategoryListResponse\$validateJSON(): Validate JSON input with respect to CategoryListResponse and throw an exception if invalid

Usage:

CategoryListResponse\$validateJSON(input)

Arguments:

input the JSON input

CategoryListResponse\$toString(): To string (JSON format)

Usage:

CategoryListResponse\$toString()

Returns: String representation of CategoryListResponse

CategoryListResponse\$isValid(): Return true if the values in all fields are valid.

Usage:

CategoryListResponse\$isValid()

Returns: true if the values in all fields are valid.

CategoryListResponse\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

CategoryListResponse\$getInvalidFields()

Returns: A list of invalid fields (if any).

CategoryListResponse\$print(): Print the object

Usage:

CategoryListResponse\$print()

CategoryListResponse\$clone(): The objects of this class are cloneable with this method.

Usage:

CategoryListResponse\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `CategoryListResponse$toList()`
## -----

# convert array of CategoryListResponse (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

Client

High-level Sybilion API client.

Description

Wraps the generated 'DefaultApi' client with base URL resolution and Bearer authentication.

Methods

Public methods:

- `Client$new()`
- `Client$get_me()`
- `Client$list_categories()`
- `Client$list_regions()`
- `Client$recommend_drivers()`
- `Client$submit_forecast()`
- `Client$get_forecast()`
- `Client$download_artifact()`
- `Client$list_jobs()`
- `Client$list_usage()`
- `Client$get_health()`
- `Client$list_all_jobs()`
- `Client$list_all_usage()`
- `Client$wait_forecast()`
- `Client$iter_usage_pages()`
- `Client$iter_jobs_pages()`
- `Client$clone()`

`Client$new():`

Usage:

```
Client$new(  
  token = NULL,  
  base_url = NULL,  
  api_key = NULL,  
  api_url = NULL,  
  user_agent = "sybilion-r/0.1.0"  
)
```

Arguments:

token Bearer credential (API key or session token). Use 'api_key' as an alias.

base_url Optional API origin; use 'api_url' as an alias.

api_key Alias for 'token' (backward compatibility).

api_url Alias for 'base_url' (backward compatibility).

user_agent Optional user agent string. Get the current authenticated user and account balances. List all available thematic categories. List all available geographic regions. Recommend driver datasets for a given timeseries.

Client\$get_me():

Usage:

```
Client$get_me()
```

Client\$list_categories():

Usage:

```
Client$list_categories()
```

Client\$list_regions():

Usage:

```
Client$list_regions()
```

Client\$recommend_drivers():

Usage:

```
Client$recommend_drivers(request)
```

Arguments:

request A RecommendRequestV1 object. Submit a forecast job.

Client\$submit_forecast():

Usage:

```
Client$submit_forecast(request)
```

Arguments:

request A ForecastRequestV1 object. Get the status of a forecast job.

Client\$get_forecast():

Usage:

```
Client$get_forecast(id)
```

Arguments:

id Forecast job UUID. Download a forecast artifact to a local file.

Client\$download_artifact():

Usage:

Client\$download_artifact(id, name, path)

Arguments:

id Forecast job UUID.

name Artifact name (e.g. "forecast.json").

path Local file path to write the artifact to. List async jobs for the current user (one page).

Client\$list_jobs():

Usage:

```
Client$list_jobs(  
  page = 1,  
  limit = 50,  
  sort = "created_at",  
  order = "desc",  
  status = NULL,  
  pipeline_type = NULL  
)
```

Arguments:

page Page number (1-indexed).

limit Results per page (1-200).

sort Sort column.

order "asc" or "desc".

status Optional status filter.

pipeline_type Optional pipeline type filter. List usage/billing events for the current user (one page).

Client\$list_usage():

Usage:

Client\$list_usage(page = 1, limit = 50, sort = "id", order = "desc")

Arguments:

page Page number (1-indexed).

limit Results per page (1-200).

sort Sort column.

order "asc" or "desc". Check API health. Fetch all async jobs across all pages.

Client\$get_health():

Usage:

Client\$get_health()

Client\$list_all_jobs():

Usage:

```
Client$list_all_jobs(  
    sort = "created_at",  
    order = "desc",  
    status = NULL,  
    pipeline_type = NULL  
)
```

Arguments:

sort Sort column.

order "asc" or "desc".

status Optional status filter.

pipeline_type Optional pipeline type filter. Fetch all usage/billing events across all pages.

```
Client$list_all_usage():
```

Usage:

```
Client$list_all_usage(sort = "id", order = "desc")
```

Arguments:

sort Sort column.

order "asc" or "desc". Poll a forecast job until it is settled or the timeout is reached.

```
Client$wait_forecast():
```

Usage:

```
Client$wait_forecast(job_id, poll_s = 2, timeout_s = 3600)
```

Arguments:

job_id Forecast job UUID.

poll_s Poll interval in seconds.

timeout_s Timeout in seconds. Iterate usage pages with a callback (same contract as Go/Python iterators).

```
Client$iter_usage_pages():
```

Usage:

```
Client$iter_usage_pages(callback, sort = "id", order = "desc", limit = 50)
```

Arguments:

callback Function called once for each page response. Iterate job pages with a callback.

```
Client$iter_jobs_pages():
```

Usage:

```
Client$iter_jobs_pages(  
    callback,  
    sort = "created_at",  
    order = "desc",  
    limit = 50,  
    status = NULL,  
    pipeline_type = NULL  
)
```

Arguments:

callback Function called once for each page response.

Client\$clone(): The objects of this class are cloneable with this method.

Usage:

```
Client$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

DefaultApi

Default operations

Description

DefaultApi

Format

An R6Class generator object

Details

Sybilion API

The Sybilion API powers the Sybilion Developers Portal: forecasts, drivers, catalog, account and usage. Authenticate every request with 'Authorization: Bearer <token>' using either an API key created in the Developers Portal or an Auth0 access token from your dashboard session.

The version of the OpenAPI document: 0.1.0 Generated by: <https://openapi-generator.tech>

Public fields

api_client Handles the client-server communication.

Methods**Public methods:**

- [DefaultApi\\$new\(\)](#)
- [DefaultApi\\$ApiV1AlertsPost\(\)](#)
- [DefaultApi\\$ApiV1AlertsPostWithHttpInfo\(\)](#)
- [DefaultApi\\$ApiV1CategoriesGet\(\)](#)
- [DefaultApi\\$ApiV1CategoriesGetWithHttpInfo\(\)](#)
- [DefaultApi\\$ApiV1DriversPost\(\)](#)
- [DefaultApi\\$ApiV1DriversPostWithHttpInfo\(\)](#)
- [DefaultApi\\$ApiV1ForecastsIdArtifactsNameGet\(\)](#)
- [DefaultApi\\$ApiV1ForecastsIdArtifactsNameGetWithHttpInfo\(\)](#)

- `DefaultApi$ApiV1ForecastsIdGet()`
- `DefaultApi$ApiV1ForecastsIdGetWithHttpInfo()`
- `DefaultApi$ApiV1ForecastsPost()`
- `DefaultApi$ApiV1ForecastsPostWithHttpInfo()`
- `DefaultApi$ApiV1JobsGet()`
- `DefaultApi$ApiV1JobsGetWithHttpInfo()`
- `DefaultApi$ApiV1MeGet()`
- `DefaultApi$ApiV1MeGetWithHttpInfo()`
- `DefaultApi$ApiV1RegionsGet()`
- `DefaultApi$ApiV1RegionsGetWithHttpInfo()`
- `DefaultApi$ApiV1UsageGet()`
- `DefaultApi$ApiV1UsageGetWithHttpInfo()`
- `DefaultApi$HealthGet()`
- `DefaultApi$HealthGetWithHttpInfo()`
- `DefaultApi$clone()`

`DefaultApi$new()`: Initialize a new DefaultApi.

Usage:

```
DefaultApi$new(api_client)
```

Arguments:

`api_client` An instance of API client.

`DefaultApi$ApiV1AlertsPost()`: Detect anomaly alerts for your timeseries

Usage:

```
DefaultApi$ApiV1AlertsPost(alerts_request_v1, data_file = NULL, ...)
```

Arguments:

`data_file` (optional) name of the data file to save the result

`...` Other optional arguments

Returns: `ApiV1AlertsPost200Response`

`DefaultApi$ApiV1AlertsPostWithHttpInfo()`: Detect anomaly alerts for your timeseries

Usage:

```
DefaultApi$ApiV1AlertsPostWithHttpInfo(
  alerts_request_v1,
  data_file = NULL,
  ...
)
```

Arguments:

`data_file` (optional) name of the data file to save the result

`...` Other optional arguments

Returns: API response (`ApiV1AlertsPost200Response`) with additional information such as HTTP status code, headers

DefaultApi\$ApiV1CategoriesGet(): List available thematic categories

Usage:

DefaultApi\$ApiV1CategoriesGet(data_file = NULL, ...)

Arguments:

data_file (optional) name of the data file to save the result

... Other optional arguments

Returns: CategoryListResponse

DefaultApi\$ApiV1CategoriesGetWithHttpInfo(): List available thematic categories

Usage:

DefaultApi\$ApiV1CategoriesGetWithHttpInfo(data_file = NULL, ...)

Arguments:

data_file (optional) name of the data file to save the result

... Other optional arguments

Returns: API response (CategoryListResponse) with additional information such as HTTP status code, headers

DefaultApi\$ApiV1DriversPost(): Rank driver datasets for your timeseries

Usage:

DefaultApi\$ApiV1DriversPost(recommend_request_v1, data_file = NULL, ...)

Arguments:

data_file (optional) name of the data file to save the result

... Other optional arguments

Returns: ApiV1DriversPost200Response

DefaultApi\$ApiV1DriversPostWithHttpInfo(): Rank driver datasets for your timeseries

Usage:

```
DefaultApi$ApiV1DriversPostWithHttpInfo(
  recommend_request_v1,
  data_file = NULL,
  ...
)
```

Arguments:

data_file (optional) name of the data file to save the result

... Other optional arguments

Returns: API response (ApiV1DriversPost200Response) with additional information such as HTTP status code, headers

DefaultApi\$ApiV1ForecastsIdArtifactsNameGet(): Download a forecast output file

Usage:

DefaultApi\$ApiV1ForecastsIdArtifactsNameGet(id, name, data_file = NULL, ...)

Arguments:

id Forecast job id.
 name Artifact filename (e.g. 'forecast.json').
 data_file (optional) name of the data file to save the result
 ... Other optional arguments

Returns: data.frame

DefaultApi\$ApiV1ForecastsIdArtifactsNameGetWithHttpInfo(): Download a forecast output file

Usage:

```
DefaultApi$ApiV1ForecastsIdArtifactsNameGetWithHttpInfo(
  id,
  name,
  data_file = NULL,
  ...
)
```

Arguments:

id Forecast job id.
 name Artifact filename (e.g. 'forecast.json').
 data_file (optional) name of the data file to save the result
 ... Other optional arguments

Returns: API response (data.frame) with additional information such as HTTP status code, headers

DefaultApi\$ApiV1ForecastsIdGet(): Poll forecast job status and artifact list

Usage:

```
DefaultApi$ApiV1ForecastsIdGet(id, data_file = NULL, ...)
```

Arguments:

id Forecast job id returned by 'POST /api/v1/forecasts'.
 data_file (optional) name of the data file to save the result
 ... Other optional arguments

Returns: ApiV1ForecastsIdGet200Response

DefaultApi\$ApiV1ForecastsIdGetWithHttpInfo(): Poll forecast job status and artifact list

Usage:

```
DefaultApi$ApiV1ForecastsIdGetWithHttpInfo(id, data_file = NULL, ...)
```

Arguments:

id Forecast job id returned by 'POST /api/v1/forecasts'.
 data_file (optional) name of the data file to save the result
 ... Other optional arguments

Returns: API response (ApiV1ForecastsIdGet200Response) with additional information such as HTTP status code, headers

DefaultApi\$ApiV1ForecastsPost(): Submit an async forecast job

Usage:

DefaultApi\$ApiV1ForecastsPost(forecast_request_v1, data_file = NULL, ...)

Arguments:

data_file (optional) name of the data file to save the result
 ... Other optional arguments

Returns: ApiV1ForecastsPost202Response

DefaultApi\$ApiV1ForecastsPostWithHttpInfo(): Submit an async forecast job

Usage:

```
DefaultApi$ApiV1ForecastsPostWithHttpInfo(
  forecast_request_v1,
  data_file = NULL,
  ...
)
```

Arguments:

data_file (optional) name of the data file to save the result
 ... Other optional arguments

Returns: API response (ApiV1ForecastsPost202Response) with additional information such as HTTP status code, headers

DefaultApi\$ApiV1JobsGet(): List your async jobs

Usage:

```
DefaultApi$ApiV1JobsGet(
  page = 1,
  limit = 50,
  sort = "created_at",
  order = "desc",
  status = NULL,
  pipeline_type = NULL,
  data_file = NULL,
  ...
)
```

Arguments:

page (optional) 1-indexed page number. (default value: 1)
 limit (optional) Page size. Capped at 200. (default value: 50)
 sort (optional) Column to sort by. (default value: "created_at")
 order (optional) Sort direction. (default value: "desc")
 status (optional) Filter to jobs in this status.
 pipeline_type (optional) Filter to jobs of this pipeline type (currently only 'forecast' is emitted).
 data_file (optional) name of the data file to save the result
 ... Other optional arguments

Returns: ApiV1JobsGet200Response

DefaultApi\$ApiV1JobsGetWithHttpInfo(): List your async jobs

Usage:

```
DefaultApi$ApiV1JobsGetWithHttpInfo(
  page = 1,
  limit = 50,
  sort = "created_at",
  order = "desc",
  status = NULL,
  pipeline_type = NULL,
  data_file = NULL,
  ...
)
```

Arguments:

page (optional) 1-indexed page number. (default value: 1)
 limit (optional) Page size. Capped at 200. (default value: 50)
 sort (optional) Column to sort by. (default value: "created_at")
 order (optional) Sort direction. (default value: "desc")
 status (optional) Filter to jobs in this status.
 pipeline_type (optional) Filter to jobs of this pipeline type (currently only 'forecast' is emitted).
 data_file (optional) name of the data file to save the result
 ... Other optional arguments

Returns: API response (ApiV1JobsGet200Response) with additional information such as HTTP status code, headers

DefaultApi\$ApiV1MeGet(): Account snapshot — balance, tier, and credit tranches

Usage:

```
DefaultApi$ApiV1MeGet(data_file = NULL, ...)
```

Arguments:

data_file (optional) name of the data file to save the result
 ... Other optional arguments

Returns: MeResponse

DefaultApi\$ApiV1MeGetWithHttpInfo(): Account snapshot — balance, tier, and credit tranches

Usage:

```
DefaultApi$ApiV1MeGetWithHttpInfo(data_file = NULL, ...)
```

Arguments:

data_file (optional) name of the data file to save the result
 ... Other optional arguments

Returns: API response (MeResponse) with additional information such as HTTP status code, headers

DefaultApi\$ApiV1RegionsGet(): List available geographic regions

Usage:

```
DefaultApi$ApiV1RegionsGet(data_file = NULL, ...)
```

Arguments:

data_file (optional) name of the data file to save the result

... Other optional arguments

Returns: RegionListResponse

DefaultApi\$ApiV1RegionsGetWithHttpInfo(): List available geographic regions

Usage:

```
DefaultApi$ApiV1RegionsGetWithHttpInfo(data_file = NULL, ...)
```

Arguments:

data_file (optional) name of the data file to save the result

... Other optional arguments

Returns: API response (RegionListResponse) with additional information such as HTTP status code, headers

DefaultApi\$ApiV1UsageGet(): Billing history (paginated usage events)

Usage:

```
DefaultApi$ApiV1UsageGet(
  page = 1,
  limit = 50,
  sort = "id",
  order = "desc",
  data_file = NULL,
  ...
)
```

Arguments:

page (optional) 1-indexed page number. (default value: 1)

limit (optional) Page size. Capped at 200. (default value: 50)

sort (optional) Column to sort by. (default value: "id")

order (optional) Sort direction. (default value: "desc")

data_file (optional) name of the data file to save the result

... Other optional arguments

Returns: ApiV1UsageGet200Response

DefaultApi\$ApiV1UsageGetWithHttpInfo(): Billing history (paginated usage events)

Usage:

```
DefaultApi$ApiV1UsageGetWithHttpInfo(
  page = 1,
  limit = 50,
  sort = "id",
  order = "desc",
  data_file = NULL,
  ...
)
```

Arguments:

page (optional) 1-indexed page number. (default value: 1)
 limit (optional) Page size. Capped at 200. (default value: 50)
 sort (optional) Column to sort by. (default value: "id")
 order (optional) Sort direction. (default value: "desc")
 data_file (optional) name of the data file to save the result
 ... Other optional arguments

Returns: API response (ApiV1UsageGet200Response) with additional information such as HTTP status code, headers

DefaultApi\$HealthGet(): Service health check

Usage:

DefaultApi\$HealthGet(data_file = NULL, ...)

Arguments:

data_file (optional) name of the data file to save the result
 ... Other optional arguments

Returns: HealthGet200Response

DefaultApi\$HealthGetWithHttpInfo(): Service health check

Usage:

DefaultApi\$HealthGetWithHttpInfo(data_file = NULL, ...)

Arguments:

data_file (optional) name of the data file to save the result
 ... Other optional arguments

Returns: API response (HealthGet200Response) with additional information such as HTTP status code, headers

DefaultApi\$clone(): The objects of this class are cloneable with this method.

Usage:

DefaultApi\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## Not run:
api <- DefaultApi$new()
api$api_client$bearer_token <- Sys.getenv("BEARER_TOKEN")
api$HealthGet()

## End(Not run)
```

DEFAULT_PUBLIC_API_BASE_URL
<i>Default public Sybillion API base URL when no explicit URL or environment override is set.</i>

Description

Default public Sybillion API base URL when no explicit URL or environment override is set.

Usage

DEFAULT_PUBLIC_API_BASE_URL

DriverItemV1	<i>DriverItemV1</i>
--------------	---------------------

Description

A ranked dataset candidate returned by 'POST /api/v1/drivers'.
DriverItemV1 Class

Format

An R6Class generator object

Details

Create a new DriverItemV1

Public fields

driver_name Human-readable name of the dataset. character [optional]
hash_id Stable identifier for the dataset; use to reference this driver across requests. character [optional]
score Relevance score indicating how well this dataset explains your timeseries (higher is more relevant). numeric [optional]

Methods

Public methods:

- [DriverItemV1\\$new\(\)](#)
- [DriverItemV1\\$toList\(\)](#)
- [DriverItemV1\\$toSimpleType\(\)](#)
- [DriverItemV1\\$fromJSON\(\)](#)

- `DriverItemV1$toJSONString()`
- `DriverItemV1$fromJSONString()`
- `DriverItemV1$validateJSON()`
- `DriverItemV1$toString()`
- `DriverItemV1$isValid()`
- `DriverItemV1$getInvalidFields()`
- `DriverItemV1$print()`
- `DriverItemV1$clone()`

`DriverItemV1$new()`: Initialize a new `DriverItemV1` class.

Usage:

```
DriverItemV1$new(driver_name = NULL, hash_id = NULL, score = NULL, ...)
```

Arguments:

`driver_name` Human-readable name of the dataset.

`hash_id` Stable identifier for the dataset; use to reference this driver across requests.

`score` Relevance score indicating how well this dataset explains your timeseries (higher is more relevant).

... Other optional arguments.

`DriverItemV1$toList()`: Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
DriverItemV1$toList()
```

Returns: `DriverItemV1` as a base R list.

Examples:

```
# convert array of DriverItemV1 (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

`DriverItemV1$toSimpleType()`: Convert `DriverItemV1` to a base R type

Usage:

```
DriverItemV1$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

`DriverItemV1$fromJSON()`: Deserialize JSON string into an instance of `DriverItemV1`

Usage:

```
DriverItemV1$fromJSON(input_json)
```

Arguments:

`input_json` the JSON input

Returns: the instance of `DriverItemV1`

DriverItemV1\$toJSONString(): To JSON String

Usage:

DriverItemV1\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: DriverItemV1 in JSON format

DriverItemV1\$fromJSONString(): Deserialize JSON string into an instance of DriverItemV1

Usage:

DriverItemV1\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of DriverItemV1

DriverItemV1\$validateJSON(): Validate JSON input with respect to DriverItemV1 and throw an exception if invalid

Usage:

DriverItemV1\$validateJSON(input)

Arguments:

input the JSON input

DriverItemV1\$toString(): To string (JSON format)

Usage:

DriverItemV1\$toString()

Returns: String representation of DriverItemV1

DriverItemV1\$isValid(): Return true if the values in all fields are valid.

Usage:

DriverItemV1\$isValid()

Returns: true if the values in all fields are valid.

DriverItemV1\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

DriverItemV1\$getInvalidFields()

Returns: A list of invalid fields (if any).

DriverItemV1\$print(): Print the object

Usage:

DriverItemV1\$print()

DriverItemV1\$clone(): The objects of this class are cloneable with this method.

Usage:

DriverItemV1\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----  
## Method `DriverItemV1$toList()`  
## -----  
  
# convert array of DriverItemV1 (x) to a data frame  
## Not run:  
library(purrr)  
library(tibble)  
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()  
df  
  
## End(Not run)
```

ErrorMessage	<i>ErrorMessage</i>
--------------	---------------------

Description

ErrorMessage Class
ErrorMessage Class

Format

An R6Class generator object

Details

Create a new ErrorMessage

Public fields

error Human-readable error message. character

Methods

Public methods:

- [ErrorMessage\\$new\(\)](#)
- [ErrorMessage\\$toList\(\)](#)
- [ErrorMessage\\$toSimpleType\(\)](#)
- [ErrorMessage\\$fromJSON\(\)](#)
- [ErrorMessage\\$toJSONString\(\)](#)
- [ErrorMessage\\$fromJSONString\(\)](#)
- [ErrorMessage\\$validateJSON\(\)](#)
- [ErrorMessage\\$toString\(\)](#)
- [ErrorMessage\\$isValid\(\)](#)

- `ErrorMessage$getInvalidFields()`
- `ErrorMessage$print()`
- `ErrorMessage$clone()`

`ErrorMessage$new()`: Initialize a new ErrorMessage class.

Usage:

```
ErrorMessage$new(error, ...)
```

Arguments:

`error` Human-readable error message.

`...` Other optional arguments.

`ErrorMessage$toList()`: Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
ErrorMessage$toList()
```

Returns: ErrorMessage as a base R list.

Examples:

```
# convert array of ErrorMessage (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

`ErrorMessage$toSimpleType()`: Convert ErrorMessage to a base R type

Usage:

```
ErrorMessage$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

`ErrorMessage$fromJSON()`: Deserialize JSON string into an instance of ErrorMessage

Usage:

```
ErrorMessage$fromJSON(input_json)
```

Arguments:

`input_json` the JSON input

Returns: the instance of ErrorMessage

`ErrorMessage$toJSONString()`: To JSON String

Usage:

```
ErrorMessage$toJSONString(...)
```

Arguments:

`...` Parameters passed to 'jsonlite::toJSON'

Returns: ErrorMessage in JSON format

ErrorMessage\$fromJsonString(): Deserialize JSON string into an instance of ErrorMessage

Usage:

ErrorMessage\$fromJsonString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ErrorMessage

ErrorMessage\$validateJSON(): Validate JSON input with respect to ErrorMessage and throw an exception if invalid

Usage:

ErrorMessage\$validateJSON(input)

Arguments:

input the JSON input

ErrorMessage\$toString(): To string (JSON format)

Usage:

ErrorMessage\$toString()

Returns: String representation of ErrorMessage

ErrorMessage\$isValid(): Return true if the values in all fields are valid.

Usage:

ErrorMessage\$isValid()

Returns: true if the values in all fields are valid.

ErrorMessage\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

ErrorMessage\$getInvalidFields()

Returns: A list of invalid fields (if any).

ErrorMessage\$print(): Print the object

Usage:

ErrorMessage\$print()

ErrorMessage\$clone(): The objects of this class are cloneable with this method.

Usage:

ErrorMessage\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `ErrorMessage$toList()`
## -----

# convert array of ErrorMessage (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

EuroTranche	<i>EuroTranche</i>
-------------	--------------------

Description

EuroTranche Class
EuroTranche Class

Format

An R6Class generator object

Details

Create a new EuroTranche

Public fields

created_at character
expires_at character
id character
initial_eur_cents integer
remaining_eur_cents integer
source One of signup_trial, stripe, partner, legacy. Other labels may appear for custom grants.
character

Methods

Public methods:

- `EuroTranche$new()`
- `EuroTranche$toList()`
- `EuroTranche$toSimpleType()`
- `EuroTranche$fromJSON()`
- `EuroTranche$toJSONString()`
- `EuroTranche$fromJSONString()`
- `EuroTranche$validateJSON()`
- `EuroTranche$toString()`
- `EuroTranche$isValid()`
- `EuroTranche$getInvalidFields()`
- `EuroTranche$print()`
- `EuroTranche$clone()`

`EuroTranche$new()`: Initialize a new EuroTranche class.

Usage:

```
EuroTranche$new(
  created_at,
  expires_at,
  id,
  initial_eur_cents,
  remaining_eur_cents,
  source,
  ...
)
```

Arguments:

`created_at` `created_at`

`expires_at` `expires_at`

`id` `id`

`initial_eur_cents` `initial_eur_cents`

`remaining_eur_cents` `remaining_eur_cents`

`source` One of `signup_trial`, `stripe`, `partner`, `legacy`. Other labels may appear for custom grants.

... Other optional arguments.

`EuroTranche$toList()`: Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
EuroTranche$toList()
```

Returns: EuroTranche as a base R list.

Examples:

```
# convert array of EuroTranche (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

EuroTranche\$toSimpleType(): Convert EuroTranche to a base R type

Usage:

```
EuroTranche$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

EuroTranche\$fromJSON(): Deserialize JSON string into an instance of EuroTranche

Usage:

```
EuroTranche$fromJSON(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of EuroTranche

EuroTranche\$toJSONString(): To JSON String

Usage:

```
EuroTranche$toJSONString(...)
```

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: EuroTranche in JSON format

EuroTranche\$fromJSONString(): Deserialize JSON string into an instance of EuroTranche

Usage:

```
EuroTranche$fromJSONString(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of EuroTranche

EuroTranche\$validateJSON(): Validate JSON input with respect to EuroTranche and throw an exception if invalid

Usage:

```
EuroTranche$validateJSON(input)
```

Arguments:

input the JSON input

EuroTranche\$toString(): To string (JSON format)

Usage:

```
EuroTranche$toString()
```

Returns: String representation of EuroTranche

`EuroTranche$isValid()`: Return true if the values in all fields are valid.

Usage:

`EuroTranche$isValid()`

Returns: true if the values in all fields are valid.

`EuroTranche$getInvalidFields()`: Return a list of invalid fields (if any).

Usage:

`EuroTranche$getInvalidFields()`

Returns: A list of invalid fields (if any).

`EuroTranche$print()`: Print the object

Usage:

`EuroTranche$print()`

`EuroTranche$clone()`: The objects of this class are cloneable with this method.

Usage:

`EuroTranche$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Examples

```
## -----
## Method `EuroTranche$toList()`
## -----

# convert array of EuroTranche (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

Filters

Filters

Description

Optional narrowing for forecast, drivers, and alerts requests. Category and region ids must fall in `**1-9999**`. Discover valid ids via `'GET /api/v1/regions'` and `'GET /api/v1/categories'` — submitted ids are not cross-checked on submit.

Filters Class

Format

An R6Class generator object

Details

Create a new Filters

Public fields

categories Thematic category ids to filter by; each must be an integer ****1–9999**** inclusive. `list(integer)` [optional]

limit Maximum number of items to return. When omitted, a per-environment default is applied (100 by default). The maximum accepted value is operator-configurable (default 1000). `integer` [optional]

regions Geographic region ids to filter by; each must be an integer ****1–9999**** inclusive. `list(integer)` [optional]

Methods**Public methods:**

- `Filters$new()`
- `Filters$toList()`
- `Filters$toSimpleType()`
- `Filters$fromJSON()`
- `Filters$toJSONString()`
- `Filters$fromJSONString()`
- `Filters$validateJSON()`
- `Filters$toString()`
- `Filters$isValid()`
- `Filters$getInvalidFields()`
- `Filters$print()`
- `Filters$clone()`

`Filters$new()`: Initialize a new Filters class.

Usage:

```
Filters$new(categories = NULL, limit = NULL, regions = NULL, ...)
```

Arguments:

categories Thematic category ids to filter by; each must be an integer ****1–9999**** inclusive.

limit Maximum number of items to return. When omitted, a per-environment default is applied (100 by default). The maximum accepted value is operator-configurable (default 1000).

regions Geographic region ids to filter by; each must be an integer ****1–9999**** inclusive.

... Other optional arguments.

Filters\$toList(): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
Filters$toList()
```

Returns: Filters as a base R list.

Examples:

```
# convert array of Filters (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

Filters\$toSimpleType(): Convert Filters to a base R type

Usage:

```
Filters$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

Filters\$fromJSON(): Deserialize JSON string into an instance of Filters

Usage:

```
Filters$fromJSON(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of Filters

Filters\$toJSONString(): To JSON String

Usage:

```
Filters$toJSONString(...)
```

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: Filters in JSON format

Filters\$fromJSONString(): Deserialize JSON string into an instance of Filters

Usage:

```
Filters$fromJSONString(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of Filters

Filters\$validateJSON(): Validate JSON input with respect to Filters and throw an exception if invalid

Usage:

```
Filters$validateJSON(input)
```

Arguments:

input the JSON input

`Filters$string()`: To string (JSON format)

Usage:

`Filters$string()`

Returns: String representation of Filters

`Filters$isValid()`: Return true if the values in all fields are valid.

Usage:

`Filters$isValid()`

Returns: true if the values in all fields are valid.

`Filters$getInvalidFields()`: Return a list of invalid fields (if any).

Usage:

`Filters$getInvalidFields()`

Returns: A list of invalid fields (if any).

`Filters$print()`: Print the object

Usage:

`Filters$print()`

`Filters$clone()`: The objects of this class are cloneable with this method.

Usage:

`Filters$clone(deep = FALSE)`

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `Filters$list()`
## -----

# convert array of Filters (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$list()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

ForecastArtifactMeta	<i>ForecastArtifactMeta</i>
----------------------	-----------------------------

Description

Metadata for a single output file produced by a completed forecast job.

ForecastArtifactMeta Class

Format

An R6Class generator object

Details

Create a new ForecastArtifactMeta

Public fields

content_type MIME type of the artifact (e.g. 'application/json'). character [optional]

href Relative URL to stream via 'GET /api/v1/forecasts/<id>/artifacts/<name>'. character [optional]

name Artifact filename (e.g. 'forecast.json', 'backtest_metrics.json'). character [optional]

size File size in bytes. integer [optional]

Methods

Public methods:

- [ForecastArtifactMeta\\$new\(\)](#)
- [ForecastArtifactMeta\\$toList\(\)](#)
- [ForecastArtifactMeta\\$toSimpleType\(\)](#)
- [ForecastArtifactMeta\\$fromJSON\(\)](#)
- [ForecastArtifactMeta\\$toJSONString\(\)](#)
- [ForecastArtifactMeta\\$fromJSONString\(\)](#)
- [ForecastArtifactMeta\\$validateJSON\(\)](#)
- [ForecastArtifactMeta\\$toString\(\)](#)
- [ForecastArtifactMeta\\$isValid\(\)](#)
- [ForecastArtifactMeta\\$getInvalidFields\(\)](#)
- [ForecastArtifactMeta\\$print\(\)](#)
- [ForecastArtifactMeta\\$clone\(\)](#)

ForecastArtifactMeta\$new(): Initialize a new ForecastArtifactMeta class.

Usage:

```
ForecastArtifactMeta$new(
  content_type = NULL,
  href = NULL,
  name = NULL,
  size = NULL,
  ...
)
```

Arguments:

`content_type` MIME type of the artifact (e.g. 'application/json').
`href` Relative URL to stream via 'GET /api/v1/forecasts/<id>/artifacts/<name>'.
`name` Artifact filename (e.g. 'forecast.json', 'backtest_metrics.json').
`size` File size in bytes.
 ... Other optional arguments.

ForecastArtifactMeta\$toList(): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
ForecastArtifactMeta$toList()
```

Returns: ForecastArtifactMeta as a base R list.

Examples:

```
# convert array of ForecastArtifactMeta (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

ForecastArtifactMeta\$toSimpleType(): Convert ForecastArtifactMeta to a base R type

Usage:

```
ForecastArtifactMeta$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

ForecastArtifactMeta\$fromJSON(): Deserialize JSON string into an instance of ForecastArtifactMeta

Usage:

```
ForecastArtifactMeta$fromJSON(input_json)
```

Arguments:

`input_json` the JSON input

Returns: the instance of ForecastArtifactMeta

ForecastArtifactMeta\$toJSONString(): To JSON String

Usage:

```
ForecastArtifactMeta$toJSONString(...)
```

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: ForecastArtifactMeta in JSON format

ForecastArtifactMeta\$fromJSONString(): Deserialize JSON string into an instance of ForecastArtifactMeta

Usage:

ForecastArtifactMeta\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ForecastArtifactMeta

ForecastArtifactMeta\$validateJSON(): Validate JSON input with respect to ForecastArtifactMeta and throw an exception if invalid

Usage:

ForecastArtifactMeta\$validateJSON(input)

Arguments:

input the JSON input

ForecastArtifactMeta\$toString(): To string (JSON format)

Usage:

ForecastArtifactMeta\$toString()

Returns: String representation of ForecastArtifactMeta

ForecastArtifactMeta\$isValid(): Return true if the values in all fields are valid.

Usage:

ForecastArtifactMeta\$isValid()

Returns: true if the values in all fields are valid.

ForecastArtifactMeta\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

ForecastArtifactMeta\$getInvalidFields()

Returns: A list of invalid fields (if any).

ForecastArtifactMeta\$print(): Print the object

Usage:

ForecastArtifactMeta\$print()

ForecastArtifactMeta\$clone(): The objects of this class are cloneable with this method.

Usage:

ForecastArtifactMeta\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `ForecastArtifactMeta$toList()`
## -----

# convert array of ForecastArtifactMeta (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

ForecastRequestV1

ForecastRequestV1

Description

Body of 'POST /api/v1/forecasts'. Submit a monthly timeseries and the pipeline produces a forward forecast (and optionally a backtest). At least one of 'soft_horizon' or 'hard_horizon' must be present. The timeseries must contain at least 60 monthly observations (5 years) aligned to the first of each month (YYYY-MM-01). 'recency_factor' controls how strongly the driver-selection step weights recent data.

ForecastRequestV1 Class

Format

An R6Class generator object

Details

Create a new ForecastRequestV1

Public fields

backtest When true, run a backtest evaluation alongside the forecast and include 'backtest_metrics.json' and 'backtest_trajectories.json' in the artifacts. character [optional]

filters Optional. Each **'categories[]'** and **'regions[]'** entry must be an integer **1–9999** (inclusive). Optional **'limit'** is **0–1000** (default **100** when omitted). Values are not verified against catalog APIs. [Filters](#) [optional]

frequency Series cadence. Only 'monthly' is currently supported. character

hard_horizon Minimum acceptable horizon (months) for the quality step-down ladder. When omitted, the pipeline falls back to a driverless forecast at 'soft_horizon' if no quality run succeeds. When still failing at 'hard_horizon', the pipeline emits a driverless forecast at that horizon. At least one of 'soft_horizon' or 'hard_horizon' must be present. When both are set, 'hard_horizon' must be strictly less than 'soft_horizon'. Maximum 12. integer [optional]

pipeline_version Pipeline version. Closed set — only 'v1' is supported today. character

recency_factor Weight given to more recent observations when selecting drivers. 0.0 = equal weight across the full history; 1.0 = strongest recency bias. numeric

soft_horizon Ideal forecast horizon (months). The pipeline tries this first, then steps down by one month until it reaches 'hard_horizon' (when set) while seeking a quality forecast. At least one of 'soft_horizon' or 'hard_horizon' must be present. When both are set, 'hard_horizon' must be strictly less than 'soft_horizon'. Maximum 12. integer [optional]

strictly_positive When true, every value in 'timeseries' must be ' ≥ 0 '; a single negative observation rejects the request with 422. The pipeline also clamps output values at zero. Defaults to false. character [optional]

timeseries Map of YYYY-MM-DD date keys to numeric observation values. Must contain at least 60 monthly observations (5 years of history) aligned to the first of each month. named list(numeric)

timeseries_metadata Describes the series so the pipeline can identify relevant drivers. [Time-seriesMetadata](#)

Methods

Public methods:

- [ForecastRequestV1\\$new\(\)](#)
- [ForecastRequestV1\\$toList\(\)](#)
- [ForecastRequestV1\\$toSimpleType\(\)](#)
- [ForecastRequestV1\\$fromJSON\(\)](#)
- [ForecastRequestV1\\$toJSONString\(\)](#)
- [ForecastRequestV1\\$fromJSONString\(\)](#)
- [ForecastRequestV1\\$validateJSON\(\)](#)
- [ForecastRequestV1\\$toString\(\)](#)
- [ForecastRequestV1\\$isValid\(\)](#)
- [ForecastRequestV1\\$getInvalidFields\(\)](#)
- [ForecastRequestV1\\$print\(\)](#)
- [ForecastRequestV1\\$clone\(\)](#)

ForecastRequestV1\$new(): Initialize a new ForecastRequestV1 class.

Usage:

```
ForecastRequestV1$new(
  frequency,
  pipeline_version,
  recency_factor,
  timeseries,
  timeseries_metadata,
  backtest = NULL,
  filters = NULL,
  hard_horizon = NULL,
  soft_horizon = NULL,
  strictly_positive = FALSE,
```

```
...
)
```

Arguments:

frequency Series cadence. Only ‘monthly’ is currently supported.

pipeline_version Pipeline version. Closed set — only ‘v1’ is supported today.

recency_factor Weight given to more recent observations when selecting drivers. 0.0 = equal weight across the full history; 1.0 = strongest recency bias.

timeseries Map of YYYY-MM-DD date keys to numeric observation values. Must contain at least 60 monthly observations (5 years of history) aligned to the first of each month.

timeseries_metadata Describes the series so the pipeline can identify relevant drivers.

backtest When true, run a backtest evaluation alongside the forecast and include ‘backtest_metrics.json’ and ‘backtest_trajectories.json’ in the artifacts.

filters Optional. Each `categories[]` and `regions[]` entry must be an integer `1-9999` (inclusive). Optional `limit` is `0-1000` (default `100` when omitted). Values are not verified against catalog APIs.

hard_horizon Minimum acceptable horizon (months) for the quality step-down ladder. When omitted, the pipeline falls back to a driverless forecast at ‘soft_horizon’ if no quality run succeeds. When still failing at ‘hard_horizon’, the pipeline emits a driverless forecast at that horizon. At least one of ‘soft_horizon’ or ‘hard_horizon’ must be present. When both are set, ‘hard_horizon’ must be strictly less than ‘soft_horizon’. Maximum 12.

soft_horizon Ideal forecast horizon (months). The pipeline tries this first, then steps down by one month until it reaches ‘hard_horizon’ (when set) while seeking a quality forecast. At least one of ‘soft_horizon’ or ‘hard_horizon’ must be present. When both are set, ‘hard_horizon’ must be strictly less than ‘soft_horizon’. Maximum 12.

strictly_positive When true, every value in ‘timeseries’ must be ‘>= 0’; a single negative observation rejects the request with 422. The pipeline also clamps output values at zero. Defaults to false.. Default to FALSE.

... Other optional arguments.

ForecastRequestV1\$toList(): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
ForecastRequestV1$toList()
```

Returns: ForecastRequestV1 as a base R list.

Examples:

```
# convert array of ForecastRequestV1 (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

ForecastRequestV1\$toSimpleType(): Convert ForecastRequestV1 to a base R type

Usage:

```
ForecastRequestV1$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

`ForecastRequestV1$fromJSON()`: Deserialize JSON string into an instance of `ForecastRequestV1`

Usage:

`ForecastRequestV1$fromJSON(input_json)`

Arguments:

`input_json` the JSON input

Returns: the instance of `ForecastRequestV1`

`ForecastRequestV1$toJSONString()`: To JSON String

Usage:

`ForecastRequestV1$toJSONString(...)`

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: `ForecastRequestV1` in JSON format

`ForecastRequestV1$fromJSONString()`: Deserialize JSON string into an instance of `ForecastRequestV1`

Usage:

`ForecastRequestV1$fromJSONString(input_json)`

Arguments:

`input_json` the JSON input

Returns: the instance of `ForecastRequestV1`

`ForecastRequestV1$validateJSON()`: Validate JSON input with respect to `ForecastRequestV1` and throw an exception if invalid

Usage:

`ForecastRequestV1$validateJSON(input)`

Arguments:

`input` the JSON input

`ForecastRequestV1$toString()`: To string (JSON format)

Usage:

`ForecastRequestV1$toString()`

Returns: String representation of `ForecastRequestV1`

`ForecastRequestV1$isValid()`: Return true if the values in all fields are valid.

Usage:

`ForecastRequestV1$isValid()`

Returns: true if the values in all fields are valid.

ForecastRequestV1\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

ForecastRequestV1\$getInvalidFields()

Returns: A list of invalid fields (if any).

ForecastRequestV1\$print(): Print the object

Usage:

ForecastRequestV1\$print()

ForecastRequestV1\$clone(): The objects of this class are cloneable with this method.

Usage:

ForecastRequestV1\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `ForecastRequestV1$toList()`
## -----

# convert array of ForecastRequestV1 (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

HealthGet200Response *HealthGet200Response*

Description

HealthGet200Response Class

HealthGet200Response Class

Format

An R6Class generator object

Details

Create a new HealthGet200Response

Public fields

status Overall health — `"ok"` when all components are healthy, `"degraded"` otherwise. character [optional]

components Per-component status map. Keys are functional names (e.g. `data_storage`, `work-flow_engine`). named list([HealthGet200ResponseComponentsValue](#)) [optional]

Methods**Public methods:**

- [HealthGet200Response\\$new\(\)](#)
- [HealthGet200Response\\$toList\(\)](#)
- [HealthGet200Response\\$toSimpleType\(\)](#)
- [HealthGet200Response\\$fromJSON\(\)](#)
- [HealthGet200Response\\$toJSONString\(\)](#)
- [HealthGet200Response\\$fromJSONString\(\)](#)
- [HealthGet200Response\\$validateJSON\(\)](#)
- [HealthGet200Response\\$toString\(\)](#)
- [HealthGet200Response\\$isValid\(\)](#)
- [HealthGet200Response\\$getInvalidFields\(\)](#)
- [HealthGet200Response\\$print\(\)](#)
- [HealthGet200Response\\$clone\(\)](#)

HealthGet200Response\$new(): Initialize a new `HealthGet200Response` class.

Usage:

```
HealthGet200Response$new(status = NULL, components = NULL, ...)
```

Arguments:

status Overall health — `"ok"` when all components are healthy, `"degraded"` otherwise.

components Per-component status map. Keys are functional names (e.g. `data_storage`, `work-flow_engine`).

... Other optional arguments.

HealthGet200Response\$toList(): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
HealthGet200Response$toList()
```

Returns: `HealthGet200Response` as a base R list.

Examples:

```
# convert array of HealthGet200Response (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

HealthGet200Response\$toSimpleType(): Convert `HealthGet200Response` to a base R type

Usage:

HealthGet200Response\$toSimpleType()

Returns: A base R type, e.g. a list or numeric/character array.

HealthGet200Response\$fromJSON(): Deserialize JSON string into an instance of HealthGet200Response

Usage:

HealthGet200Response\$fromJSON(input_json)

Arguments:

input_json the JSON input

Returns: the instance of HealthGet200Response

HealthGet200Response\$toJSONString(): To JSON String

Usage:

HealthGet200Response\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: HealthGet200Response in JSON format

HealthGet200Response\$fromJSONString(): Deserialize JSON string into an instance of HealthGet200Response

Usage:

HealthGet200Response\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of HealthGet200Response

HealthGet200Response\$validateJSON(): Validate JSON input with respect to HealthGet200Response and throw an exception if invalid

Usage:

HealthGet200Response\$validateJSON(input)

Arguments:

input the JSON input

HealthGet200Response\$toString(): To string (JSON format)

Usage:

HealthGet200Response\$toString()

Returns: String representation of HealthGet200Response

HealthGet200Response\$isValid(): Return true if the values in all fields are valid.

Usage:

HealthGet200Response\$isValid()

Returns: true if the values in all fields are valid.

HealthGet200Response\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

```
HealthGet200Response$getInvalidFields()
```

Returns: A list of invalid fields (if any).

HealthGet200Response\$print(): Print the object

Usage:

```
HealthGet200Response$print()
```

HealthGet200Response\$clone(): The objects of this class are cloneable with this method.

Usage:

```
HealthGet200Response$clone(deep = FALSE)
```

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `HealthGet200Response$toList()`
## -----

# convert array of HealthGet200Response (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

HealthGet200ResponseComponentsValue

HealthGet200ResponseComponentsValue

Description

HealthGet200ResponseComponentsValue Class

HealthGet200ResponseComponentsValue Class

Format

An R6Class generator object

Details

Create a new HealthGet200ResponseComponentsValue

Public fields

status character [optional]

Methods**Public methods:**

- `HealthGet200ResponseComponentsValue$new()`
- `HealthGet200ResponseComponentsValue$toList()`
- `HealthGet200ResponseComponentsValue$toSimpleType()`
- `HealthGet200ResponseComponentsValue$fromJSON()`
- `HealthGet200ResponseComponentsValue$toJSONString()`
- `HealthGet200ResponseComponentsValue$fromJSONString()`
- `HealthGet200ResponseComponentsValue$validateJSON()`
- `HealthGet200ResponseComponentsValue$toString()`
- `HealthGet200ResponseComponentsValue$isValid()`
- `HealthGet200ResponseComponentsValue$getInvalidFields()`
- `HealthGet200ResponseComponentsValue$print()`
- `HealthGet200ResponseComponentsValue$clone()`

`HealthGet200ResponseComponentsValue$new()`: Initialize a new `HealthGet200ResponseComponentsValue` class.

Usage:

```
HealthGet200ResponseComponentsValue$new(status = NULL, ...)
```

Arguments:

status status

... Other optional arguments.

`HealthGet200ResponseComponentsValue$toList()`: Convert to a List
Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
HealthGet200ResponseComponentsValue$toList()
```

Returns: `HealthGet200ResponseComponentsValue` as a base R list.

Examples:

```
# convert array of HealthGet200ResponseComponentsValue (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

`HealthGet200ResponseComponentsValue$toSimpleType()`: Convert `HealthGet200ResponseComponentsValue` to a base R type

Usage:

```
HealthGet200ResponseComponentsValue$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

HealthGet200ResponseComponentsValue\$fromJSON(): Deserialize JSON string into an instance of HealthGet200ResponseComponentsValue

Usage:

HealthGet200ResponseComponentsValue\$fromJSON(input_json)

Arguments:

input_json the JSON input

Returns: the instance of HealthGet200ResponseComponentsValue

HealthGet200ResponseComponentsValue\$toJSONString(): To JSON String

Usage:

HealthGet200ResponseComponentsValue\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: HealthGet200ResponseComponentsValue in JSON format

HealthGet200ResponseComponentsValue\$fromJSONString(): Deserialize JSON string into an instance of HealthGet200ResponseComponentsValue

Usage:

HealthGet200ResponseComponentsValue\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of HealthGet200ResponseComponentsValue

HealthGet200ResponseComponentsValue\$validateJSON(): Validate JSON input with respect to HealthGet200ResponseComponentsValue and throw an exception if invalid

Usage:

HealthGet200ResponseComponentsValue\$validateJSON(input)

Arguments:

input the JSON input

HealthGet200ResponseComponentsValue\$toString(): To string (JSON format)

Usage:

HealthGet200ResponseComponentsValue\$toString()

Returns: String representation of HealthGet200ResponseComponentsValue

HealthGet200ResponseComponentsValue\$isValid(): Return true if the values in all fields are valid.

Usage:

HealthGet200ResponseComponentsValue\$isValid()

Returns: true if the values in all fields are valid.

HealthGet200ResponseComponentsValue\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

HealthGet200ResponseComponentsValue\$getInvalidFields()

Returns: A list of invalid fields (if any).

HealthGet200ResponseComponentsValue\$print(): Print the object

Usage:

HealthGet200ResponseComponentsValue\$print()

HealthGet200ResponseComponentsValue\$clone(): The objects of this class are cloneable with this method.

Usage:

HealthGet200ResponseComponentsValue\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `HealthGet200ResponseComponentsValue$toList()`
## -----

# convert array of HealthGet200ResponseComponentsValue (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

HealthGet503Response *HealthGet503Response*

Description

HealthGet503Response Class

HealthGet503Response Class

Format

An R6Class generator object

Details

Create a new HealthGet503Response

Public fields

status character [optional]

components named list([HealthGet503ResponseComponentsValue](#)) [optional]

Methods**Public methods:**

- [HealthGet503Response\\$new\(\)](#)
- [HealthGet503Response\\$toList\(\)](#)
- [HealthGet503Response\\$toSimpleType\(\)](#)
- [HealthGet503Response\\$fromJSON\(\)](#)
- [HealthGet503Response\\$toJSONString\(\)](#)
- [HealthGet503Response\\$fromJSONString\(\)](#)
- [HealthGet503Response\\$validateJSON\(\)](#)
- [HealthGet503Response\\$toString\(\)](#)
- [HealthGet503Response\\$isValid\(\)](#)
- [HealthGet503Response\\$getInvalidFields\(\)](#)
- [HealthGet503Response\\$print\(\)](#)
- [HealthGet503Response\\$clone\(\)](#)

[HealthGet503Response\\$new\(\)](#): Initialize a new [HealthGet503Response](#) class.

Usage:

```
HealthGet503Response$new(status = NULL, components = NULL, ...)
```

Arguments:

status status

components components

... Other optional arguments.

[HealthGet503Response\\$toList\(\)](#): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
HealthGet503Response$toList()
```

Returns: [HealthGet503Response](#) as a base R list.

Examples:

```
# convert array of HealthGet503Response (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

[HealthGet503Response\\$toSimpleType\(\)](#): Convert [HealthGet503Response](#) to a base R type

Usage:

```
HealthGet503Response$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

HealthGet503Response\$fromJSON(): Deserialize JSON string into an instance of HealthGet503Response

Usage:

HealthGet503Response\$fromJSON(input_json)

Arguments:

input_json the JSON input

Returns: the instance of HealthGet503Response

HealthGet503Response\$toJSONString(): To JSON String

Usage:

HealthGet503Response\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: HealthGet503Response in JSON format

HealthGet503Response\$fromJSONString(): Deserialize JSON string into an instance of HealthGet503Response

Usage:

HealthGet503Response\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of HealthGet503Response

HealthGet503Response\$validateJSON(): Validate JSON input with respect to HealthGet503Response and throw an exception if invalid

Usage:

HealthGet503Response\$validateJSON(input)

Arguments:

input the JSON input

HealthGet503Response\$toString(): To string (JSON format)

Usage:

HealthGet503Response\$toString()

Returns: String representation of HealthGet503Response

HealthGet503Response\$isValid(): Return true if the values in all fields are valid.

Usage:

HealthGet503Response\$isValid()

Returns: true if the values in all fields are valid.

HealthGet503Response\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

HealthGet503Response\$getInvalidFields()

Returns: A list of invalid fields (if any).

HealthGet503Response\$print(): Print the object

Usage:

HealthGet503Response\$print()

HealthGet503Response\$clone(): The objects of this class are cloneable with this method.

Usage:

HealthGet503Response\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `HealthGet503Response$toList()`
## -----

# convert array of HealthGet503Response (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

HealthGet503ResponseComponentsValue

HealthGet503ResponseComponentsValue

Description

HealthGet503ResponseComponentsValue Class

HealthGet503ResponseComponentsValue Class

Format

An R6Class generator object

Details

Create a new HealthGet503ResponseComponentsValue

Public fields

status character [optional]

Methods**Public methods:**

- `HealthGet503ResponseComponentsValue$new()`
- `HealthGet503ResponseComponentsValue$toList()`
- `HealthGet503ResponseComponentsValue$toSimpleType()`
- `HealthGet503ResponseComponentsValue$fromJSON()`
- `HealthGet503ResponseComponentsValue$toJSONString()`
- `HealthGet503ResponseComponentsValue$fromJSONString()`
- `HealthGet503ResponseComponentsValue$validateJSON()`
- `HealthGet503ResponseComponentsValue$toString()`
- `HealthGet503ResponseComponentsValue$isValid()`
- `HealthGet503ResponseComponentsValue$getInvalidFields()`
- `HealthGet503ResponseComponentsValue$print()`
- `HealthGet503ResponseComponentsValue$clone()`

`HealthGet503ResponseComponentsValue$new()`: Initialize a new `HealthGet503ResponseComponentsValue` class.

Usage:

```
HealthGet503ResponseComponentsValue$new(status = NULL, ...)
```

Arguments:

status status

... Other optional arguments.

`HealthGet503ResponseComponentsValue$toList()`: Convert to a List
Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
HealthGet503ResponseComponentsValue$toList()
```

Returns: `HealthGet503ResponseComponentsValue` as a base R list.

Examples:

```
# convert array of HealthGet503ResponseComponentsValue (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

`HealthGet503ResponseComponentsValue$toSimpleType()`: Convert `HealthGet503ResponseComponentsValue` to a base R type

Usage:

```
HealthGet503ResponseComponentsValue$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

`HealthGet503ResponseComponentsValue$fromJSON()`: Deserialize JSON string into an instance of `HealthGet503ResponseComponentsValue`

Usage:

`HealthGet503ResponseComponentsValue$fromJSON(input_json)`

Arguments:

`input_json` the JSON input

Returns: the instance of `HealthGet503ResponseComponentsValue`

`HealthGet503ResponseComponentsValue$toJSONString()`: To JSON String

Usage:

`HealthGet503ResponseComponentsValue$toJSONString(...)`

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: `HealthGet503ResponseComponentsValue` in JSON format

`HealthGet503ResponseComponentsValue$fromJSONString()`: Deserialize JSON string into an instance of `HealthGet503ResponseComponentsValue`

Usage:

`HealthGet503ResponseComponentsValue$fromJSONString(input_json)`

Arguments:

`input_json` the JSON input

Returns: the instance of `HealthGet503ResponseComponentsValue`

`HealthGet503ResponseComponentsValue$validateJSON()`: Validate JSON input with respect to `HealthGet503ResponseComponentsValue` and throw an exception if invalid

Usage:

`HealthGet503ResponseComponentsValue$validateJSON(input)`

Arguments:

`input` the JSON input

`HealthGet503ResponseComponentsValue$toString()`: To string (JSON format)

Usage:

`HealthGet503ResponseComponentsValue$toString()`

Returns: String representation of `HealthGet503ResponseComponentsValue`

`HealthGet503ResponseComponentsValue$isValid()`: Return true if the values in all fields are valid.

Usage:

`HealthGet503ResponseComponentsValue$isValid()`

Returns: true if the values in all fields are valid.

HealthGet503ResponseComponentsValue\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

HealthGet503ResponseComponentsValue\$getInvalidFields()

Returns: A list of invalid fields (if any).

HealthGet503ResponseComponentsValue\$print(): Print the object

Usage:

HealthGet503ResponseComponentsValue\$print()

HealthGet503ResponseComponentsValue\$clone(): The objects of this class are cloneable with this method.

Usage:

HealthGet503ResponseComponentsValue\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `HealthGet503ResponseComponentsValue$toList()`
## -----

# convert array of HealthGet503ResponseComponentsValue (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

JobsPagination	JobsPagination
----------------	----------------

Description

JobsPagination Class
JobsPagination Class

Format

An R6Class generator object

Details

Create a new JobsPagination

Public fields

`limit` Page size echoed back from the request. integer
`order` Sort direction echoed back from the request. character
`page` 1-indexed current page number echoed back from the request. integer
`sort` Column the rows are sorted by, echoed back from the request. character
`total` Total matching rows for the authenticated user (full set, not just this page). integer
`total_pages` $\text{ceil}(\text{total} / \text{limit})$. Zero when total is zero. integer

Methods**Public methods:**

- `JobsPagination$new()`
- `JobsPagination$toList()`
- `JobsPagination$toSimpleType()`
- `JobsPagination$fromJSON()`
- `JobsPagination$toJSONString()`
- `JobsPagination$fromJSONString()`
- `JobsPagination$validateJSON()`
- `JobsPagination$toString()`
- `JobsPagination$isValid()`
- `JobsPagination$getInvalidFields()`
- `JobsPagination$print()`
- `JobsPagination$clone()`

`JobsPagination$new()`: Initialize a new JobsPagination class.

Usage:

`JobsPagination$new(limit, order, page, sort, total, total_pages, ...)`

Arguments:

`limit` Page size echoed back from the request.
`order` Sort direction echoed back from the request.
`page` 1-indexed current page number echoed back from the request.
`sort` Column the rows are sorted by, echoed back from the request.
`total` Total matching rows for the authenticated user (full set, not just this page).
`total_pages` $\text{ceil}(\text{total} / \text{limit})$. Zero when total is zero.
`...` Other optional arguments.

`JobsPagination$toList()`: Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

`JobsPagination$toList()`

Returns: JobsPagination as a base R list.

Examples:

```
# convert array of JobsPagination (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

JobsPagination\$toSimpleType(): Convert JobsPagination to a base R type

Usage:

```
JobsPagination$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

JobsPagination\$fromJSON(): Deserialize JSON string into an instance of JobsPagination

Usage:

```
JobsPagination$fromJSON(input_json)
```

Arguments:

`input_json` the JSON input

Returns: the instance of JobsPagination

JobsPagination\$toJSONString(): To JSON String

Usage:

```
JobsPagination$toJSONString(...)
```

Arguments:

`...` Parameters passed to 'jsonlite::toJSON'

Returns: JobsPagination in JSON format

JobsPagination\$fromJSONString(): Deserialize JSON string into an instance of JobsPagination

Usage:

```
JobsPagination$fromJSONString(input_json)
```

Arguments:

`input_json` the JSON input

Returns: the instance of JobsPagination

JobsPagination\$validateJSON(): Validate JSON input with respect to JobsPagination and throw an exception if invalid

Usage:

```
JobsPagination$validateJSON(input)
```

Arguments:

`input` the JSON input

JobsPagination\$toString(): To string (JSON format)

Usage:

`JobsPagination$toString()`

Returns: String representation of JobsPagination

`JobsPagination$isValid()`: Return true if the values in all fields are valid.

Usage:

`JobsPagination$isValid()`

Returns: true if the values in all fields are valid.

`JobsPagination$getInvalidFields()`: Return a list of invalid fields (if any).

Usage:

`JobsPagination$getInvalidFields()`

Returns: A list of invalid fields (if any).

`JobsPagination$print()`: Print the object

Usage:

`JobsPagination$print()`

`JobsPagination$clone()`: The objects of this class are cloneable with this method.

Usage:

`JobsPagination$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Examples

```
## -----
## Method `JobsPagination$toList()`
## -----

# convert array of JobsPagination (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

JobSummary

JobSummary

Description

Lightweight projection of an `async_jobs` row (no payload, no manifest).

JobSummary Class

Format

An R6Class generator object

Details

Create a new JobSummary

Public fields

`created_at` character

`eur_cents_final` Final settled charge for the job, in EUR cents. Null until settled. integer [optional]

`job_id` character

`pipeline_type` Today only `"forecast"`; future pipeline types will appear here. character

`run_id` Internal run id (omitted for jobs that haven't started yet). character [optional]

`settled` True iff the job has reached a terminal state and been billed. character

`settled_at` character [optional]

`status` character

`terminal_reason` character [optional]

`workflow_id` Internal workflow id (omitted for jobs that haven't started yet). character [optional]

Methods**Public methods:**

- `JobSummary$new()`
- `JobSummary$toList()`
- `JobSummary$toSimpleType()`
- `JobSummary$fromJSON()`
- `JobSummary$toJSONString()`
- `JobSummary$fromJSONString()`
- `JobSummary$validateJSON()`
- `JobSummary$toString()`
- `JobSummary$isValid()`
- `JobSummary$getInvalidFields()`

- `JobSummary$print()`
- `JobSummary$clone()`

`JobSummary$new()`: Initialize a new `JobSummary` class.

Usage:

```
JobSummary$new(
  created_at,
  job_id,
  pipeline_type,
  settled,
  status,
  eur_cents_final = NULL,
  run_id = NULL,
  settled_at = NULL,
  terminal_reason = NULL,
  workflow_id = NULL,
  ...
)
```

Arguments:

`created_at` `created_at`
`job_id` `job_id`
`pipeline_type` Today only `"forecast"`; future pipeline types will appear here.
`settled` True iff the job has reached a terminal state and been billed.
`status` `status`
`eur_cents_final` Final settled charge for the job, in EUR cents. Null until settled.
`run_id` Internal run id (omitted for jobs that haven't started yet).
`settled_at` `settled_at`
`terminal_reason` `terminal_reason`
`workflow_id` Internal workflow id (omitted for jobs that haven't started yet).
`...` Other optional arguments.

`JobSummary$toList()`: Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
JobSummary$toList()
```

Returns: `JobSummary` as a base R list.

Examples:

```
# convert array of JobSummary (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

`JobSummary$toSimpleType()`: Convert `JobSummary` to a base R type

Usage:

`JobSummary$toSimpleType()`

Returns: A base R type, e.g. a list or numeric/character array.

`JobSummary$fromJSON()`: Deserialize JSON string into an instance of JobSummary

Usage:

`JobSummary$fromJSON(input_json)`

Arguments:

`input_json` the JSON input

Returns: the instance of JobSummary

`JobSummary$toJSONString()`: To JSON String

Usage:

`JobSummary$toJSONString(...)`

Arguments:

`...` Parameters passed to 'jsonlite::toJSON'

Returns: JobSummary in JSON format

`JobSummary$fromJSONString()`: Deserialize JSON string into an instance of JobSummary

Usage:

`JobSummary$fromJSONString(input_json)`

Arguments:

`input_json` the JSON input

Returns: the instance of JobSummary

`JobSummary$validateJSON()`: Validate JSON input with respect to JobSummary and throw an exception if invalid

Usage:

`JobSummary$validateJSON(input)`

Arguments:

`input` the JSON input

`JobSummary$toString()`: To string (JSON format)

Usage:

`JobSummary$toString()`

Returns: String representation of JobSummary

`JobSummary$isValid()`: Return true if the values in all fields are valid.

Usage:

`JobSummary$isValid()`

Returns: true if the values in all fields are valid.

`JobSummary$getInvalidFields()`: Return a list of invalid fields (if any).

Usage:

`JobSummary$getInvalidFields()`

Returns: A list of invalid fields (if any).

`JobSummary$print()`: Print the object

Usage:

`JobSummary$print()`

`JobSummary$clone()`: The objects of this class are cloneable with this method.

Usage:

`JobSummary$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Examples

```
## -----
## Method `JobSummary$toList()`
## -----

# convert array of JobSummary (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

MeResponse

MeResponse

Description

Authenticated user snapshot. All monetary fields are integer EUR cents (1 EUR = 100 cents).

MeResponse Class

Format

An R6Class generator object

Details

Create a new MeResponse

Public fields

api_usage_tier Current pricing tier level. Higher levels unlock better rate limits and job concurrency. integer
 auto_recharge [AutoRechargeState](#) [optional]
 available_eur_cents Spendable balance in EUR cents — ‘balance_eur_cents’ minus any credits held for in-flight async jobs. integer
 balance_eur_cents Total credit balance in EUR cents, before deducting active holds. integer
 euro_tranches Active credit grants (non-empty, unexpired), consumed in ‘expires_at’ ascending order. list([EuroTranche](#))
 has_ever_paid True once the account has completed at least one successful Stripe payment. character
 lifetime_paid_cents Cumulative EUR cents charged across all Stripe payments, all time. integer
 payment_count Total number of successful Stripe payments on this account. integer
 signup_trial [MeResponseSignupTrial](#) [optional]
 user_id character

Methods**Public methods:**

- [MeResponse\\$new\(\)](#)
- [MeResponse\\$toList\(\)](#)
- [MeResponse\\$toSimpleType\(\)](#)
- [MeResponse\\$fromJSON\(\)](#)
- [MeResponse\\$toJSONString\(\)](#)
- [MeResponse\\$fromJSONString\(\)](#)
- [MeResponse\\$validateJSON\(\)](#)
- [MeResponse\\$toString\(\)](#)
- [MeResponse\\$isValid\(\)](#)
- [MeResponse\\$getInvalidFields\(\)](#)
- [MeResponse\\$print\(\)](#)
- [MeResponse\\$clone\(\)](#)

MeResponse\$new(): Initialize a new MeResponse class.

Usage:

```
MeResponse$new(
  api_usage_tier,
  available_eur_cents,
  balance_eur_cents,
  euro_tranches,
  has_ever_paid,
  lifetime_paid_cents,
  payment_count,
```

```

    user_id,
    auto_recharge = NULL,
    signup_trial = NULL,
    ...
)

```

Arguments:

`api_usage_tier` Current pricing tier level. Higher levels unlock better rate limits and job concurrency.

`available_eur_cents` Spendable balance in EUR cents — ‘balance_eur_cents’ minus any credits held for in-flight async jobs.

`balance_eur_cents` Total credit balance in EUR cents, before deducting active holds.

`euro_tranches` Active credit grants (non-empty, unexpired), consumed in ‘expires_at’ ascending order.

`has_ever_paid` True once the account has completed at least one successful Stripe payment.

`lifetime_paid_cents` Cumulative EUR cents charged across all Stripe payments, all time.

`payment_count` Total number of successful Stripe payments on this account.

`user_id` `user_id`

`auto_recharge` `auto_recharge`

`signup_trial` `signup_trial`

... Other optional arguments.

`MeResponse$toList()`: Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
MeResponse$toList()
```

Returns: MeResponse as a base R list.

Examples:

```

# convert array of MeResponse (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

```

`MeResponse$toSimpleType()`: Convert MeResponse to a base R type

Usage:

```
MeResponse$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

`MeResponse$fromJSON()`: Deserialize JSON string into an instance of MeResponse

Usage:

```
MeResponse$fromJSON(input_json)
```

Arguments:

`input_json` the JSON input

Returns: the instance of MeResponse

MeResponse\$toJSONString(): To JSON String

Usage:

MeResponse\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: MeResponse in JSON format

MeResponse\$fromJSONString(): Deserialize JSON string into an instance of MeResponse

Usage:

MeResponse\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of MeResponse

MeResponse\$validateJSON(): Validate JSON input with respect to MeResponse and throw an exception if invalid

Usage:

MeResponse\$validateJSON(input)

Arguments:

input the JSON input

MeResponse\$toString(): To string (JSON format)

Usage:

MeResponse\$toString()

Returns: String representation of MeResponse

MeResponse\$isValid(): Return true if the values in all fields are valid.

Usage:

MeResponse\$isValid()

Returns: true if the values in all fields are valid.

MeResponse\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

MeResponse\$getInvalidFields()

Returns: A list of invalid fields (if any).

MeResponse\$print(): Print the object

Usage:

MeResponse\$print()

MeResponse\$clone(): The objects of this class are cloneable with this method.

Usage:

MeResponse\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `MeResponse$toList()`
## -----

# convert array of MeResponse (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

MeResponseSignupTrial *MeResponseSignupTrial*

Description

Present when a signup (free) trial tranche was granted. Omitted for users with no trial or before grant.

MeResponseSignupTrial Class

Format

An R6Class generator object

Details

Create a new MeResponseSignupTrial

Public fields

expires_at Omitted if the grant row is missing (abnormal) character [optional]
 granted_at character [optional]
 initial_eur_cents integer [optional]
 remaining_eur_cents integer [optional]

Methods**Public methods:**

- [MeResponseSignupTrial\\$new\(\)](#)
- [MeResponseSignupTrial\\$toList\(\)](#)
- [MeResponseSignupTrial\\$toSimpleType\(\)](#)
- [MeResponseSignupTrial\\$fromJSON\(\)](#)
- [MeResponseSignupTrial\\$toJSONString\(\)](#)

- `MeResponseSignupTrial$fromJSONString()`
- `MeResponseSignupTrial$validateJSON()`
- `MeResponseSignupTrial$toString()`
- `MeResponseSignupTrial$isValid()`
- `MeResponseSignupTrial$getInvalidFields()`
- `MeResponseSignupTrial$print()`
- `MeResponseSignupTrial$clone()`

`MeResponseSignupTrial$new()`: Initialize a new `MeResponseSignupTrial` class.

Usage:

```
MeResponseSignupTrial$new(
  expires_at = NULL,
  granted_at = NULL,
  initial_eur_cents = NULL,
  remaining_eur_cents = NULL,
  ...
)
```

Arguments:

`expires_at` Omitted if the grant row is missing (abnormal)
`granted_at` `granted_at`
`initial_eur_cents` `initial_eur_cents`
`remaining_eur_cents` `remaining_eur_cents`
 ... Other optional arguments.

`MeResponseSignupTrial$toList()`: Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
MeResponseSignupTrial$toList()
```

Returns: `MeResponseSignupTrial` as a base R list.

Examples:

```
# convert array of MeResponseSignupTrial (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

`MeResponseSignupTrial$toSimpleType()`: Convert `MeResponseSignupTrial` to a base R type

Usage:

```
MeResponseSignupTrial$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

`MeResponseSignupTrial$fromJSON()`: Deserialize JSON string into an instance of `MeResponseSignupTrial`

Usage:

MeResponseSignupTrial\$fromJSON(input_json)

Arguments:

input_json the JSON input

Returns: the instance of MeResponseSignupTrial

MeResponseSignupTrial\$toJSONString(): To JSON String

Usage:

MeResponseSignupTrial\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: MeResponseSignupTrial in JSON format

MeResponseSignupTrial\$fromJSONString(): Deserialize JSON string into an instance of MeResponseSignupTrial

Usage:

MeResponseSignupTrial\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of MeResponseSignupTrial

MeResponseSignupTrial\$validateJSON(): Validate JSON input with respect to MeResponseSignupTrial and throw an exception if invalid

Usage:

MeResponseSignupTrial\$validateJSON(input)

Arguments:

input the JSON input

MeResponseSignupTrial\$toString(): To string (JSON format)

Usage:

MeResponseSignupTrial\$toString()

Returns: String representation of MeResponseSignupTrial

MeResponseSignupTrial\$isValid(): Return true if the values in all fields are valid.

Usage:

MeResponseSignupTrial\$isValid()

Returns: true if the values in all fields are valid.

MeResponseSignupTrial\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

MeResponseSignupTrial\$getInvalidFields()

Returns: A list of invalid fields (if any).

MeResponseSignupTrial\$print(): Print the object

Usage:

MeResponseSignupTrial\$print()

MeResponseSignupTrial\$clone(): The objects of this class are cloneable with this method.

Usage:

MeResponseSignupTrial\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----  
## Method `MeResponseSignupTrial$toList()`  
## -----  
  
# convert array of MeResponseSignupTrial (x) to a data frame  
## Not run:  
library(purrr)  
library(tibble)  
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()  
df  
  
## End(Not run)
```

NewsItemV1	NewsItemV1
------------	------------

Description

A news article associated with an alert.

NewsItemV1 Class

Format

An R6Class generator object

Details

Create a new NewsItemV1

Public fields

category Topical category of the article (e.g. 'world', 'business', 'energy'). character [optional]
description Short summary of the article. character [optional]
published_at Publication timestamp (RFC 3339 / ISO 8601). character [optional]
source_name Name of the publication or media outlet. character [optional]
title Headline of the news article. character [optional]
trending Whether this article is currently trending across the platform. character [optional]
url Canonical URL of the article. character [optional]

Methods**Public methods:**

- `NewsItemV1$new()`
- `NewsItemV1$list()`
- `NewsItemV1$toSimpleType()`
- `NewsItemV1$fromJSON()`
- `NewsItemV1$toJSONString()`
- `NewsItemV1$fromJSONString()`
- `NewsItemV1$validateJSON()`
- `NewsItemV1$toString()`
- `NewsItemV1$isValid()`
- `NewsItemV1$getInvalidFields()`
- `NewsItemV1$print()`
- `NewsItemV1$clone()`

`NewsItemV1$new()`: Initialize a new `NewsItemV1` class.

Usage:

```
NewsItemV1$new(  
  category = NULL,  
  description = NULL,  
  published_at = NULL,  
  source_name = NULL,  
  title = NULL,  
  trending = NULL,  
  url = NULL,  
  ...  
)
```

Arguments:

category Topical category of the article (e.g. 'world', 'business', 'energy').
description Short summary of the article.
published_at Publication timestamp (RFC 3339 / ISO 8601).
source_name Name of the publication or media outlet.
title Headline of the news article.

trending Whether this article is currently trending across the platform.
 url Canonical URL of the article.
 ... Other optional arguments.

NewsItemV1\$toList(): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
NewsItemV1$toList()
```

Returns: NewsItemV1 as a base R list.

Examples:

```
# convert array of NewsItemV1 (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

NewsItemV1\$toSimpleType(): Convert NewsItemV1 to a base R type

Usage:

```
NewsItemV1$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

NewsItemV1\$fromJSON(): Deserialize JSON string into an instance of NewsItemV1

Usage:

```
NewsItemV1$fromJSON(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of NewsItemV1

NewsItemV1\$toJSONString(): To JSON String

Usage:

```
NewsItemV1$toJSONString(...)
```

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: NewsItemV1 in JSON format

NewsItemV1\$fromJSONString(): Deserialize JSON string into an instance of NewsItemV1

Usage:

```
NewsItemV1$fromJSONString(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of NewsItemV1

NewsItemV1\$validateJSON(): Validate JSON input with respect to NewsItemV1 and throw an exception if invalid

Usage:

NewsItemV1\$validateJSON(input)

Arguments:

input the JSON input

NewsItemV1\$string(): To string (JSON format)

Usage:

NewsItemV1\$string()

Returns: String representation of NewsItemV1

NewsItemV1\$isValid(): Return true if the values in all fields are valid.

Usage:

NewsItemV1\$isValid()

Returns: true if the values in all fields are valid.

NewsItemV1\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

NewsItemV1\$getInvalidFields()

Returns: A list of invalid fields (if any).

NewsItemV1\$print(): Print the object

Usage:

NewsItemV1\$print()

NewsItemV1\$clone(): The objects of this class are cloneable with this method.

Usage:

NewsItemV1\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `NewsItemV1$list()`
## -----

# convert array of NewsItemV1 (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$list()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

Pagination	<i>Pagination</i>
------------	-------------------

Description

Pagination Class
Pagination Class

Format

An R6Class generator object

Details

Create a new Pagination

Public fields

limit Page size echoed back from the request. integer
order Sort direction. character
page 1-indexed current page number echoed back from the request. integer
sort Column the rows are sorted by. character
total Total matching rows for the authenticated user (full set, not just this page). integer
total_pages ceil(total / limit). Zero when total is zero. integer

Methods

Public methods:

- `Pagination$new()`
- `Pagination$toList()`
- `Pagination$toSimpleType()`
- `Pagination$fromJSON()`
- `Pagination$toJSONString()`
- `Pagination$fromJSONString()`
- `Pagination$validateJSON()`
- `Pagination$toString()`
- `Pagination$isValid()`
- `Pagination$getInvalidFields()`
- `Pagination$print()`
- `Pagination$clone()`

`Pagination$new()`: Initialize a new Pagination class.

Usage:

```
Pagination$new(limit, order, page, sort, total, total_pages, ...)
```

Arguments:

limit Page size echoed back from the request.

order Sort direction.

page 1-indexed current page number echoed back from the request.

sort Column the rows are sorted by.

total Total matching rows for the authenticated user (full set, not just this page).

total_pages $\text{ceil}(\text{total} / \text{limit})$. Zero when total is zero.

... Other optional arguments.

Pagination\$toList(): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
Pagination$toList()
```

Returns: Pagination as a base R list.

Examples:

```
# convert array of Pagination (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

Pagination\$toSimpleType(): Convert Pagination to a base R type

Usage:

```
Pagination$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

Pagination\$fromJSON(): Deserialize JSON string into an instance of Pagination

Usage:

```
Pagination$fromJSON(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of Pagination

Pagination\$toJSONString(): To JSON String

Usage:

```
Pagination$toJSONString(...)
```

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: Pagination in JSON format

Pagination\$fromJSONString(): Deserialize JSON string into an instance of Pagination

Usage:

Pagination\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of Pagination

Pagination\$validateJSON(): Validate JSON input with respect to Pagination and throw an exception if invalid

Usage:

Pagination\$validateJSON(input)

Arguments:

input the JSON input

Pagination\$toString(): To string (JSON format)

Usage:

Pagination\$toString()

Returns: String representation of Pagination

Pagination\$isValid(): Return true if the values in all fields are valid.

Usage:

Pagination\$isValid()

Returns: true if the values in all fields are valid.

Pagination\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

Pagination\$getInvalidFields()

Returns: A list of invalid fields (if any).

Pagination\$print(): Print the object

Usage:

Pagination\$print()

Pagination\$clone(): The objects of this class are cloneable with this method.

Usage:

Pagination\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `Pagination$toList()`
## -----

# convert array of Pagination (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

RecommendRequestV1	<i>RecommendRequestV1</i>
--------------------	---------------------------

Description

Body of 'POST /api/v1/drivers'. Mirrors the upstream Recommend service contract. Note that the version field is named 'version' (not 'pipeline_version' as on '/forecasts'). Required fields: 'version', 'recency_factor', 'timeseries_metadata'. Both 'filters' and 'timeseries' are optional — when omitted, the handler drops them from the upstream payload entirely (no 'null' is sent). When 'filters.limit' is omitted, a per-environment default is applied.

RecommendRequestV1 Class

Format

An R6Class generator object

Details

Create a new RecommendRequestV1

Public fields

filters Optional. Each **'categories[]'** and **'regions[]'** entry must be an integer **1–9999** (inclusive). Optional **'limit'** is **0–1000** (default **100** when omitted). Values are not verified against catalog APIs. [Filters](#) [optional]

recency_factor Weight given to more recent observations when ranking drivers. 0.0 = equal weight; 1.0 = strongest recency bias. numeric

timeseries Optional. Map of YYYY-MM-DD date keys to numeric observation values. When supplied, all keys must parse as YYYY-MM-DD and all values must be finite. Unlike '/forecasts', this endpoint is frequency-agnostic — no monthly alignment, gap detection, or minimum length is enforced. When omitted, the handler does not forward the field upstream. **named list(numeric)** [optional]

`timeseries_metadata` Describes the series so the ranking model can identify relevant drivers.
[TimeseriesMetadata](#)

`version` Recommend pipeline version. Closed set; only 'v1' is supported today. Used locally to select the per-version validator and is **not forwarded** to the upstream Recommend service.
 character

Methods

Public methods:

- [RecommendRequestV1\\$new\(\)](#)
- [RecommendRequestV1\\$toList\(\)](#)
- [RecommendRequestV1\\$toSimpleType\(\)](#)
- [RecommendRequestV1\\$fromJSON\(\)](#)
- [RecommendRequestV1\\$toJSONString\(\)](#)
- [RecommendRequestV1\\$fromJSONString\(\)](#)
- [RecommendRequestV1\\$validateJSON\(\)](#)
- [RecommendRequestV1\\$toString\(\)](#)
- [RecommendRequestV1\\$isValid\(\)](#)
- [RecommendRequestV1\\$getInvalidFields\(\)](#)
- [RecommendRequestV1\\$print\(\)](#)
- [RecommendRequestV1\\$clone\(\)](#)

`RecommendRequestV1$new()`: Initialize a new `RecommendRequestV1` class.

Usage:

```
RecommendRequestV1$new(
  recency_factor,
  timeseries_metadata,
  version,
  filters = NULL,
  timeseries = NULL,
  ...
)
```

Arguments:

`recency_factor` Weight given to more recent observations when ranking drivers. 0.0 = equal weight; 1.0 = strongest recency bias.

`timeseries_metadata` Describes the series so the ranking model can identify relevant drivers.

`version` Recommend pipeline version. Closed set; only 'v1' is supported today. Used locally to select the per-version validator and is **not forwarded** to the upstream Recommend service.

`filters` Optional. Each `'categories[]'` and `'regions[]'` entry must be an integer `1–9999` (inclusive). Optional `'limit'` is `0–1000` (default `100` when omitted). Values are not verified against catalog APIs.

`timeseries` Optional. Map of YYYY-MM-DD date keys to numeric observation values. When supplied, all keys must parse as YYYY-MM-DD and all values must be finite. Unlike `'/forecasts'`, this endpoint is frequency-agnostic — no monthly alignment, gap detection,

or minimum length is enforced. When omitted, the handler does not forward the field upstream.

... Other optional arguments.

RecommendRequestV1\$toList(): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
RecommendRequestV1$toList()
```

Returns: RecommendRequestV1 as a base R list.

Examples:

```
# convert array of RecommendRequestV1 (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

RecommendRequestV1\$toSimpleType(): Convert RecommendRequestV1 to a base R type

Usage:

```
RecommendRequestV1$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

RecommendRequestV1\$fromJSON(): Deserialize JSON string into an instance of RecommendRequestV1

Usage:

```
RecommendRequestV1$fromJSON(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of RecommendRequestV1

RecommendRequestV1\$toJSONString(): To JSON String

Usage:

```
RecommendRequestV1$toJSONString(...)
```

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: RecommendRequestV1 in JSON format

RecommendRequestV1\$fromJSONString(): Deserialize JSON string into an instance of RecommendRequestV1

Usage:

```
RecommendRequestV1$fromJSONString(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of RecommendRequestV1

`RecommendRequestV1$validateJSON()`: Validate JSON input with respect to RecommendRequestV1 and throw an exception if invalid

Usage:

`RecommendRequestV1$validateJSON(input)`

Arguments:

`input` the JSON input

`RecommendRequestV1$toString()`: To string (JSON format)

Usage:

`RecommendRequestV1$toString()`

Returns: String representation of RecommendRequestV1

`RecommendRequestV1$isValid()`: Return true if the values in all fields are valid.

Usage:

`RecommendRequestV1$isValid()`

Returns: true if the values in all fields are valid.

`RecommendRequestV1$getInvalidFields()`: Return a list of invalid fields (if any).

Usage:

`RecommendRequestV1$getInvalidFields()`

Returns: A list of invalid fields (if any).

`RecommendRequestV1$print()`: Print the object

Usage:

`RecommendRequestV1$print()`

`RecommendRequestV1$clone()`: The objects of this class are cloneable with this method.

Usage:

`RecommendRequestV1$clone(deep = FALSE)`

Arguments:

`deep` Whether to make a deep clone.

Examples

```
## -----
## Method `RecommendRequestV1$toList()`
## -----

# convert array of RecommendRequestV1 (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

RegionItemV1

RegionItemV1

Description

A single geographic region returned by GET /api/v1/regions.

RegionItemV1 Class

Format

An R6Class generator object

Details

Create a new RegionItemV1

Public fields

id Integer identifier. Use this value in filters.regions[], integer

latitude Geographic latitude (0.0 when not applicable). numeric

longitude Geographic longitude (0.0 when not applicable). numeric

name Human-readable region label. character

Methods

Public methods:

- [RegionItemV1\\$new\(\)](#)
- [RegionItemV1\\$list\(\)](#)
- [RegionItemV1\\$toSimpleType\(\)](#)
- [RegionItemV1\\$fromJSON\(\)](#)
- [RegionItemV1\\$toJSONString\(\)](#)
- [RegionItemV1\\$fromJSONString\(\)](#)
- [RegionItemV1\\$validateJSON\(\)](#)
- [RegionItemV1\\$toString\(\)](#)
- [RegionItemV1\\$isValid\(\)](#)
- [RegionItemV1\\$getInvalidFields\(\)](#)
- [RegionItemV1\\$print\(\)](#)
- [RegionItemV1\\$clone\(\)](#)

RegionItemV1\$new(): Initialize a new RegionItemV1 class.

Usage:

RegionItemV1\$new(id, latitude, longitude, name, ...)

Arguments:

id Integer identifier. Use this value in filters.regions[].
 latitude Geographic latitude (0.0 when not applicable).
 longitude Geographic longitude (0.0 when not applicable).
 name Human-readable region label.
 ... Other optional arguments.

RegionItemV1\$toList(): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
RegionItemV1$toList()
```

Returns: RegionItemV1 as a base R list.

Examples:

```
# convert array of RegionItemV1 (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

RegionItemV1\$toSimpleType(): Convert RegionItemV1 to a base R type

Usage:

```
RegionItemV1$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

RegionItemV1\$fromJSON(): Deserialize JSON string into an instance of RegionItemV1

Usage:

```
RegionItemV1$fromJSON(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of RegionItemV1

RegionItemV1\$toJSONString(): To JSON String

Usage:

```
RegionItemV1$toJSONString(...)
```

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: RegionItemV1 in JSON format

RegionItemV1\$fromJSONString(): Deserialize JSON string into an instance of RegionItemV1

Usage:

```
RegionItemV1$fromJSONString(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of RegionItemV1

RegionItemV1\$validateJSON(): Validate JSON input with respect to RegionItemV1 and throw an exception if invalid

Usage:

RegionItemV1\$validateJSON(input)

Arguments:

input the JSON input

RegionItemV1\$toString(): To string (JSON format)

Usage:

RegionItemV1\$toString()

Returns: String representation of RegionItemV1

RegionItemV1\$isValid(): Return true if the values in all fields are valid.

Usage:

RegionItemV1\$isValid()

Returns: true if the values in all fields are valid.

RegionItemV1\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

RegionItemV1\$getInvalidFields()

Returns: A list of invalid fields (if any).

RegionItemV1\$print(): Print the object

Usage:

RegionItemV1\$print()

RegionItemV1\$clone(): The objects of this class are cloneable with this method.

Usage:

RegionItemV1\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `RegionItemV1$toList()`
## -----

# convert array of RegionItemV1 (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

RegionListResponse	<i>RegionListResponse</i>
--------------------	---------------------------

Description

RegionListResponse Class

RegionListResponse Class

Format

An R6Class generator object

Details

Create a new RegionListResponse

Public fields

`items` Complete region listing, sorted by id ascending. No pagination. list([RegionItemV1](#))

Methods

Public methods:

- [RegionListResponse\\$new\(\)](#)
- [RegionListResponse\\$toList\(\)](#)
- [RegionListResponse\\$toSimpleType\(\)](#)
- [RegionListResponse\\$fromJSON\(\)](#)
- [RegionListResponse\\$toJSONString\(\)](#)
- [RegionListResponse\\$fromJSONString\(\)](#)
- [RegionListResponse\\$validateJSON\(\)](#)
- [RegionListResponse\\$toString\(\)](#)
- [RegionListResponse\\$isValid\(\)](#)
- [RegionListResponse\\$getInvalidFields\(\)](#)
- [RegionListResponse\\$print\(\)](#)
- [RegionListResponse\\$clone\(\)](#)

`RegionListResponse$new()`: Initialize a new RegionListResponse class.

Usage:

```
RegionListResponse$new(items, ...)
```

Arguments:

`items` Complete region listing, sorted by id ascending. No pagination.

`...` Other optional arguments.

`RegionListResponse$toList()`: Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
RegionListResponse$toList()
```

Returns: RegionListResponse as a base R list.

Examples:

```
# convert array of RegionListResponse (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

RegionListResponse\$toSimpleType(): Convert RegionListResponse to a base R type

Usage:

```
RegionListResponse$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

RegionListResponse\$fromJSON(): Deserialize JSON string into an instance of RegionListResponse

Usage:

```
RegionListResponse$fromJSON(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of RegionListResponse

RegionListResponse\$toJSONString(): To JSON String

Usage:

```
RegionListResponse$toJSONString(...)
```

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: RegionListResponse in JSON format

RegionListResponse\$fromJSONString(): Deserialize JSON string into an instance of RegionListResponse

Usage:

```
RegionListResponse$fromJSONString(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of RegionListResponse

RegionListResponse\$validateJSON(): Validate JSON input with respect to RegionListResponse and throw an exception if invalid

Usage:

```
RegionListResponse$validateJSON(input)
```

Arguments:

input the JSON input

RegionListResponse\$string(): To string (JSON format)

Usage:

RegionListResponse\$string()

Returns: String representation of RegionListResponse

RegionListResponse\$isValid(): Return true if the values in all fields are valid.

Usage:

RegionListResponse\$isValid()

Returns: true if the values in all fields are valid.

RegionListResponse\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

RegionListResponse\$getInvalidFields()

Returns: A list of invalid fields (if any).

RegionListResponse\$print(): Print the object

Usage:

RegionListResponse\$print()

RegionListResponse\$clone(): The objects of this class are cloneable with this method.

Usage:

RegionListResponse\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `RegionListResponse$list()`
## -----

# convert array of RegionListResponse (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$list()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

resolve_api_url	<i>Resolve the Sybillion API URL.</i>
-----------------	---------------------------------------

Description

Order: non-empty explicit, then SYBILION_API_BASE_URL, then the compiled default.

Usage

resolve_api_url(explicit = NULL)

Arguments

explicit Optional API origin.

Value

API origin without trailing slash.

SYBILION_API_BASE_URL_ENV	<i>Name of the environment variable that overrides the Sybillion API base URL (read by resolve_api_url).</i>
---------------------------	--

Description

Name of the environment variable that overrides the Sybillion API base URL (read by [resolve_api_url](#)).

Usage

SYBILION_API_BASE_URL_ENV

sybillion_client	Construct a high-level Sybillion client.
------------------	--

Description

Construct a high-level Sybillion client.

Usage

```
sybillion_client(token = NULL, base_url = NULL, api_key = NULL, api_url = NULL)
```

Arguments

- token Bearer credential. Ignored if you pass ‘api_key’ instead.
- base_url Optional API origin (alias: ‘api_url’).
- api_key Alias for ‘token’.
- api_url Alias for ‘base_url’.

Value

A ‘Client’ instance.

TimeseriesMetadata	<i>TimeseriesMetadata</i>
--------------------	---------------------------

Description

Descriptive metadata the ranking model uses to interpret and contextualize the timeseries.
TimeseriesMetadata Class

Format

An R6Class generator object

Details

Create a new TimeseriesMetadata

Public fields

- description Extended context for the model, up to 2048 characters. More detail improves driver relevance. character [optional]
- keywords Up to 20 semantic tags that help anchor the search to relevant datasets. list(character) [optional]
- title Short identifier for the series, 20–511 characters. character

Methods

Public methods:

- [TimeseriesMetadata\\$new\(\)](#)
- [TimeseriesMetadata\\$toList\(\)](#)
- [TimeseriesMetadata\\$toSimpleType\(\)](#)
- [TimeseriesMetadata\\$fromJSON\(\)](#)
- [TimeseriesMetadata\\$toJSONString\(\)](#)
- [TimeseriesMetadata\\$fromJSONString\(\)](#)
- [TimeseriesMetadata\\$validateJSON\(\)](#)
- [TimeseriesMetadata\\$toString\(\)](#)
- [TimeseriesMetadata\\$isValid\(\)](#)
- [TimeseriesMetadata\\$getInvalidFields\(\)](#)
- [TimeseriesMetadata\\$print\(\)](#)
- [TimeseriesMetadata\\$clone\(\)](#)

`TimeseriesMetadata$new()`: Initialize a new `TimeseriesMetadata` class.

Usage:

```
TimeseriesMetadata$new(title, description = NULL, keywords = NULL, ...)
```

Arguments:

`title` Short identifier for the series, 20–511 characters.

`description` Extended context for the model, up to 2048 characters. More detail improves driver relevance.

`keywords` Up to 20 semantic tags that help anchor the search to relevant datasets.

`...` Other optional arguments.

`TimeseriesMetadata$toList()`: Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
TimeseriesMetadata$toList()
```

Returns: `TimeseriesMetadata` as a base R list.

Examples:

```
# convert array of TimeseriesMetadata (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

`TimeseriesMetadata$toSimpleType()`: Convert `TimeseriesMetadata` to a base R type

Usage:

```
TimeseriesMetadata$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

TimeseriesMetadata\$fromJSON(): Deserialize JSON string into an instance of TimeseriesMetadata

Usage:

TimeseriesMetadata\$fromJSON(input_json)

Arguments:

input_json the JSON input

Returns: the instance of TimeseriesMetadata

TimeseriesMetadata\$toJSONString(): To JSON String

Usage:

TimeseriesMetadata\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: TimeseriesMetadata in JSON format

TimeseriesMetadata\$fromJSONString(): Deserialize JSON string into an instance of TimeseriesMetadata

Usage:

TimeseriesMetadata\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of TimeseriesMetadata

TimeseriesMetadata\$validateJSON(): Validate JSON input with respect to TimeseriesMetadata and throw an exception if invalid

Usage:

TimeseriesMetadata\$validateJSON(input)

Arguments:

input the JSON input

TimeseriesMetadata\$toString(): To string (JSON format)

Usage:

TimeseriesMetadata\$toString()

Returns: String representation of TimeseriesMetadata

TimeseriesMetadata\$isValid(): Return true if the values in all fields are valid.

Usage:

TimeseriesMetadata\$isValid()

Returns: true if the values in all fields are valid.

TimeseriesMetadata\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

TimeseriesMetadata\$getInvalidFields()

Returns: A list of invalid fields (if any).

TimeseriesMetadata\$print(): Print the object

Usage:

TimeseriesMetadata\$print()

TimeseriesMetadata\$clone(): The objects of this class are cloneable with this method.

Usage:

TimeseriesMetadata\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----  
## Method `TimeseriesMetadata$toList()`  
## -----  
  
# convert array of TimeseriesMetadata (x) to a data frame  
## Not run:  
library(purrr)  
library(tibble)  
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()  
df  
  
## End(Not run)
```

UsageEvent	UsageEvent
------------	------------

Description

UsageEvent Class

UsageEvent Class

Format

An R6Class generator object

Details

Create a new UsageEvent

Public fields

`async_job_id` `async_jobs.id` (UUID) for async pipeline charges; null for sync endpoint charges
 character [optional]
`created_at` Timestamp (ISO-8601 text from the database) character
`credits_charged` Whole credits debited for this usage row before EUR conversion ('eur_cents_charged').
 integer
`endpoint` Billing endpoint key (e.g. forecast pipeline type or sync route key) character [optional]
`eur_cents_charged` EUR cents debited for this usage row (1 EUR = 100 cents). integer
`id` integer
`units` Strategy-defined metering quantity (e.g. tokens, seconds); not necessarily equal to credits.
 integer

Methods**Public methods:**

- `UsageEvent$new()`
- `UsageEvent$toList()`
- `UsageEvent$toSimpleType()`
- `UsageEvent$fromJSON()`
- `UsageEvent$toJSONString()`
- `UsageEvent$fromJSONString()`
- `UsageEvent$validateJSON()`
- `UsageEvent$toString()`
- `UsageEvent$isValid()`
- `UsageEvent$getInvalidFields()`
- `UsageEvent$print()`
- `UsageEvent$clone()`

`UsageEvent$new()`: Initialize a new `UsageEvent` class.

Usage:

```
UsageEvent$new(
  created_at,
  credits_charged,
  eur_cents_charged,
  id,
  units,
  async_job_id = NULL,
  endpoint = NULL,
  ...
)
```

Arguments:

`created_at` Timestamp (ISO-8601 text from the database)
`credits_charged` Whole credits debited for this usage row before EUR conversion ('eur_cents_charged').

eur_cents_charged EUR cents debited for this usage row (1 EUR = 100 cents).
 id id
 units Strategy-defined metering quantity (e.g. tokens, seconds); not necessarily equal to credits.
 async_job_id async_jobs.id (UUID) for async pipeline charges; null for sync endpoint charges
 endpoint Billing endpoint key (e.g. forecast pipeline type or sync route key)
 ... Other optional arguments.

UsageEvent\$toList(): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
UsageEvent$toList()
```

Returns: UsageEvent as a base R list.

Examples:

```
# convert array of UsageEvent (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

UsageEvent\$toSimpleType(): Convert UsageEvent to a base R type

Usage:

```
UsageEvent$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

UsageEvent\$fromJSON(): Deserialize JSON string into an instance of UsageEvent

Usage:

```
UsageEvent$fromJSON(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of UsageEvent

UsageEvent\$toJSONString(): To JSON String

Usage:

```
UsageEvent$toJSONString(...)
```

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: UsageEvent in JSON format

UsageEvent\$fromJSONString(): Deserialize JSON string into an instance of UsageEvent

Usage:

```
UsageEvent$fromJSONString(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of UsageEvent

UsageEvent\$validateJSON(): Validate JSON input with respect to UsageEvent and throw an exception if invalid

Usage:

UsageEvent\$validateJSON(input)

Arguments:

input the JSON input

UsageEvent\$toString(): To string (JSON format)

Usage:

UsageEvent\$toString()

Returns: String representation of UsageEvent

UsageEvent\$isValid(): Return true if the values in all fields are valid.

Usage:

UsageEvent\$isValid()

Returns: true if the values in all fields are valid.

UsageEvent\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

UsageEvent\$getInvalidFields()

Returns: A list of invalid fields (if any).

UsageEvent\$print(): Print the object

Usage:

UsageEvent\$print()

UsageEvent\$clone(): The objects of this class are cloneable with this method.

Usage:

UsageEvent\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `UsageEvent$toList()`
## -----

# convert array of UsageEvent (x) to a data frame
## Not run:
```

```
library(purrr)
library(tibble)
df <- x |> map(\(y)y$list()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

ValidationErrorResponse

ValidationErrorResponse

Description

ValidationErrorResponse Class

ValidationErrorResponse Class

Format

An R6Class generator object

Details

Create a new `ValidationErrorResponse`

Public fields

details list([ValidationErrorResponseDetailsInner](#))

error character

Methods

Public methods:

- [ValidationErrorResponse\\$new\(\)](#)
- [ValidationErrorResponse\\$list\(\)](#)
- [ValidationErrorResponse\\$toSimpleType\(\)](#)
- [ValidationErrorResponse\\$fromJSON\(\)](#)
- [ValidationErrorResponse\\$toJSONString\(\)](#)
- [ValidationErrorResponse\\$fromJSONString\(\)](#)
- [ValidationErrorResponse\\$validateJSON\(\)](#)
- [ValidationErrorResponse\\$toString\(\)](#)
- [ValidationErrorResponse\\$isValid\(\)](#)
- [ValidationErrorResponse\\$getInvalidFields\(\)](#)
- [ValidationErrorResponse\\$print\(\)](#)
- [ValidationErrorResponse\\$clone\(\)](#)

`ValidationErrorResponse$new()`: Initialize a new `ValidationErrorResponse` class.

Usage:

```
ValidationErrorResponse$new(details, error, ...)
```

Arguments:

details details

error error

... Other optional arguments.

ValidationErrorResponse\$toList(): Convert to a List

Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
ValidationErrorResponse$toList()
```

Returns: ValidationErrorResponse as a base R list.

Examples:

```
# convert array of ValidationErrorResponse (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

ValidationErrorResponse\$toSimpleType(): Convert ValidationErrorResponse to a base R type

Usage:

```
ValidationErrorResponse$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

ValidationErrorResponse\$fromJSON(): Deserialize JSON string into an instance of ValidationErrorResponse

Usage:

```
ValidationErrorResponse$fromJSON(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of ValidationErrorResponse

ValidationErrorResponse\$toJSONString(): To JSON String

Usage:

```
ValidationErrorResponse$toJSONString(...)
```

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: ValidationErrorResponse in JSON format

ValidationErrorResponse\$fromJSONString(): Deserialize JSON string into an instance of ValidationErrorResponse

Usage:

ValidationErrorResponse\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ValidationErrorResponse

ValidationErrorResponse\$validateJSON(): Validate JSON input with respect to ValidationErrorResponse and throw an exception if invalid

Usage:

ValidationErrorResponse\$validateJSON(input)

Arguments:

input the JSON input

ValidationErrorResponse\$toString(): To string (JSON format)

Usage:

ValidationErrorResponse\$toString()

Returns: String representation of ValidationErrorResponse

ValidationErrorResponse\$isValid(): Return true if the values in all fields are valid.

Usage:

ValidationErrorResponse\$isValid()

Returns: true if the values in all fields are valid.

ValidationErrorResponse\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

ValidationErrorResponse\$getInvalidFields()

Returns: A list of invalid fields (if any).

ValidationErrorResponse\$print(): Print the object

Usage:

ValidationErrorResponse\$print()

ValidationErrorResponse\$clone(): The objects of this class are cloneable with this method.

Usage:

ValidationErrorResponse\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----
## Method `ValidationErrorResponse$toList()`
## -----

# convert array of ValidationErrorResponse (x) to a data frame
## Not run:
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df

## End(Not run)
```

ValidationErrorResponseDetailsInner

ValidationErrorResponseDetailsInner

Description

ValidationErrorResponseDetailsInner Class

ValidationErrorResponseDetailsInner Class

Format

An R6Class generator object

Details

Create a new ValidationErrorResponseDetailsInner

Public fields

field character

message character

Methods**Public methods:**

- [ValidationErrorResponseDetailsInner\\$new\(\)](#)
- [ValidationErrorResponseDetailsInner\\$toList\(\)](#)
- [ValidationErrorResponseDetailsInner\\$toSimpleType\(\)](#)
- [ValidationErrorResponseDetailsInner\\$fromJSON\(\)](#)
- [ValidationErrorResponseDetailsInner\\$toJSONString\(\)](#)
- [ValidationErrorResponseDetailsInner\\$fromJSONString\(\)](#)
- [ValidationErrorResponseDetailsInner\\$validateJSON\(\)](#)

- `ValidationErrorResponseDetailsInner$toString()`
- `ValidationErrorResponseDetailsInner$isValid()`
- `ValidationErrorResponseDetailsInner$getInvalidFields()`
- `ValidationErrorResponseDetailsInner$print()`
- `ValidationErrorResponseDetailsInner$clone()`

`ValidationErrorResponseDetailsInner$new()`: Initialize a new `ValidationErrorResponseDetailsInner` class.

Usage:

```
ValidationErrorResponseDetailsInner$new(field, message, ...)
```

Arguments:

field field

message message

... Other optional arguments.

`ValidationErrorResponseDetailsInner$toList()`: Convert to a List
Convert the R6 object to a list to work more easily with other tooling.

Usage:

```
ValidationErrorResponseDetailsInner$toList()
```

Returns: `ValidationErrorResponseDetailsInner` as a base R list.

Examples:

```
# convert array of ValidationErrorResponseDetailsInner (x) to a data frame
library(purrr)
library(tibble)
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()
df
```

`ValidationErrorResponseDetailsInner$toSimpleType()`: Convert `ValidationErrorResponseDetailsInner` to a base R type

Usage:

```
ValidationErrorResponseDetailsInner$toSimpleType()
```

Returns: A base R type, e.g. a list or numeric/character array.

`ValidationErrorResponseDetailsInner$fromJSON()`: Deserialize JSON string into an instance of `ValidationErrorResponseDetailsInner`

Usage:

```
ValidationErrorResponseDetailsInner$fromJSON(input_json)
```

Arguments:

input_json the JSON input

Returns: the instance of `ValidationErrorResponseDetailsInner`

`ValidationErrorResponseDetailsInner$toJSONString()`: To JSON String

Usage:

ValidationErrorResponseDetailsInner\$toJSONString(...)

Arguments:

... Parameters passed to 'jsonlite::toJSON'

Returns: ValidationErrorResponseDetailsInner in JSON format

ValidationErrorResponseDetailsInner\$fromJSONString(): Deserialize JSON string into an instance of ValidationErrorResponseDetailsInner

Usage:

ValidationErrorResponseDetailsInner\$fromJSONString(input_json)

Arguments:

input_json the JSON input

Returns: the instance of ValidationErrorResponseDetailsInner

ValidationErrorResponseDetailsInner\$validateJSON(): Validate JSON input with respect to ValidationErrorResponseDetailsInner and throw an exception if invalid

Usage:

ValidationErrorResponseDetailsInner\$validateJSON(input)

Arguments:

input the JSON input

ValidationErrorResponseDetailsInner\$toString(): To string (JSON format)

Usage:

ValidationErrorResponseDetailsInner\$toString()

Returns: String representation of ValidationErrorResponseDetailsInner

ValidationErrorResponseDetailsInner\$isValid(): Return true if the values in all fields are valid.

Usage:

ValidationErrorResponseDetailsInner\$isValid()

Returns: true if the values in all fields are valid.

ValidationErrorResponseDetailsInner\$getInvalidFields(): Return a list of invalid fields (if any).

Usage:

ValidationErrorResponseDetailsInner\$getInvalidFields()

Returns: A list of invalid fields (if any).

ValidationErrorResponseDetailsInner\$print(): Print the object

Usage:

ValidationErrorResponseDetailsInner\$print()

ValidationErrorResponseDetailsInner\$clone(): The objects of this class are cloneable with this method.

Usage:

ValidationErrorResponseDetailsInner\$clone(deep = FALSE)

Arguments:

deep Whether to make a deep clone.

Examples

```
## -----  
## Method `ValidationErrorResponseDetailsInner$toList()`  
## -----  
  
# convert array of ValidationErrorResponseDetailsInner (x) to a data frame  
## Not run:  
library(purrr)  
library(tibble)  
df <- x |> map(\(y)y$toList()) |> map(as_tibble) |> list_rbind()  
df  
  
## End(Not run)
```

Index

AlertItemV1, [2](#), [16](#)
AlertsRequestV1, [6](#)
AnyType, [9](#)
ApiClient, [10](#)
ApiResponse, [15](#)
ApiV1AlertsPost200Response, [16](#)
ApiV1DriversPost200Response, [19](#)
ApiV1DriversPost200ResponseData, [19](#), [22](#)
ApiV1ForecastsIdGet200Response, [25](#)
ApiV1ForecastsPost202Response, [29](#)
ApiV1JobsGet200Response, [32](#)
ApiV1UsageGet200Response, [35](#)
AutoRechargeState, [38](#), [100](#)

CategoryItemV1, [42](#), [45](#)
CategoryListResponse, [45](#)
Client, [48](#)

DEFAULT_PUBLIC_API_BASE_URL, [60](#)
DefaultApi, [52](#)
DriverItemV1, [23](#), [60](#)

ErrorMessage, [63](#)
EuroTranche, [66](#), [100](#)

Filters, [6](#), [69](#), [76](#), [113](#)
ForecastArtifactMeta, [26](#), [73](#)
ForecastRequestV1, [76](#)

HealthGet200Response, [80](#)
HealthGet200ResponseComponentsValue, [81](#), [83](#)
HealthGet503Response, [86](#)
HealthGet503ResponseComponentsValue, [87](#), [89](#)

JobsPagination, [33](#), [92](#)
JobSummary, [33](#), [96](#)

MeResponse, [99](#)
MeResponseSignupTrial, [100](#), [103](#)

NewsItemV1, [3](#), [106](#)

Pagination, [36](#), [110](#)

RecommendRequestV1, [113](#)
RegionItemV1, [117](#), [120](#)
RegionListResponse, [120](#)
resolve_api_url, [123](#), [123](#)

SYBILION_API_BASE_URL_ENV, [123](#)
sybillion_client, [124](#)

TimeseriesMetadata, [6](#), [77](#), [114](#), [124](#)

UsageEvent, [36](#), [127](#)

ValidationErrorResponse, [131](#)
ValidationErrorResponseDetailsInner, [131](#), [134](#)